5-10-2003

Fundamentals of Software Patent Protection at a University

Christopher E. Everett

FUNDAMENTALS OF SOFTWARE PATENT

PROTECTION AT A UNIVERSITY

By

Christopher E. Everett

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfllment of the Requirements
for the Degree of Master of Science
in Computer Science
in the Department of Computer Science

Mississippi State, Mississippi

May 2003

FUNDAMENTALS OF SOFTWARE PATENT

PROTECTION AT A UNIVERSITY

By

Christopher E. Everett

Approved:

Edward B. Allen Assistant Professor of Computer Science (Major Professor)	Jonathan W. Pote Professor of Agricultural and Biological Engineering and Vice-President of Research (Committee Member)
David A. Dampier Assistant Professor of Computer Science (Committee Member)	Robert R. Little Associate Professor of Computer Science (Committee Member)
Julian E. Boggess Associate Professor of Computer Science Graduate Coordinator Department of Computer Science	A. Wayne Bennett Dean of the College of Engineering

Name: Christopher E. Everett

Date of Degree: May 10, 2003

Institution: Mississippi State University

Major Field: Computer Science

Major Professor: Dr. Edward B. Allen

Title of Study: FUNDAMENTALS OF SOFTWARE PATENT PROTECTION AT A UNIVERSITY

Pages in Study: 69

Candidate for Degree of Master of Science

Software protection by patents is an emerging feld and thus is not completely understood by software developers, especially software developers in a university setting. University inventors have to balance their publication productivity and the desire of their university to license inventions that could be proftable. This balance stems from the one-year bar on fling a U.S. patent application after public disclosure such as publications of the invention. The research provides evidence supporting the hypothesis that a university inventor can improve the protection of his or her software patent by applying certain information about patent prosecution practices and the relevant prior art.

Software inventors need to be concerned about fulflling the requirements of patent laws. Some of the methods for fulflling these requirements include using diagrams in patent applications such as functional block diagrams, fo wchart diagrams, and state diagrams and ensuring that the patent application is understandable by non-technical people.

The knowledge of prior art ensures that the inventor is not ”reinventing the wheel,” not infringing on a patent, and understands the current state of the art. The knowledge of patent laws, diagrams, readability, and prior art enables a software inventor to take control of the protection of his or her invention to ensure that the application of this information leads to improvements during the application process.

DEDICATION

To Chrissie.

ACKNOWLEDGMENTS

I thank Chrissie for helpful discussions regarding this thesis and my committee for their comments on this thesis, and I thank Dr. Edward B. Allen for directing this research.

I thank the Mississippi State University Offce of Research and Offce of Intellectual Property and Technology Licensing for their support, especially Dr. Robert Altenkirch, Dr. Jonathan Pote, and Mr. Charles Rivenburgh for their support and guidance.

I also thank Mr. Charles Warner, Mr. James Heintz, Mr. Wilburn Chesser, and Mr. Steven Kelber for their guidance and kind encouragement in my entry into the patent profession.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

GLOSSARY

Allowed Patent A patent application that has been through the patent prosecution process of the United State Patents and Trademark Offce and is deemed novel, nonobvious, and useful [61].

Classifcation of Patent The United States Patent and Trademark Offce has divided patents into classifcations according to their feld of invention for sorting purposes.

Dependent Claim A patent claim that refers back to a preceding claim and cannot stand on its own.

Independent Claim A patent claim that can stand on its own without referring to any other claim.

Invention Disclosure A signed disclosure of an invention to a third party such as a patent practitioner or a university technology licensing offce.

One-Year Bar In the United States, there is a one-year time limit to fle a patent application after an invention is published, sold, or disclosed to the public. In most foreign countries, as soon as an invention is published, sold, or disclosed to the public all patent rights in that country are lost.

Patent Claim The language that establishes the bounds of an invention and must distinctly claim the matter that is regarded as the invention. The two types of claims are independent and dependent [61].

Patent Examiner An employee of the United States Patent and Trademark Offce who reviews applications for patents and determines whether patents can be granted [61].

Patent Practitioner An attorney or agent who is registered to represent inventors in front of the U.S. Patent and Trademark Offce. To be registered, a person must have the legal, scientifc, and technical qualifcations to adequately prosecute a patent and must pass an examination [61].

Patent Prosecution The process of guiding a patent application through the United States Patent and Trademark Offce [51].

Precedents In court, the body of knowledge from prior court cases that has priority in deciding new cases.

Prior Art The state of knowledge existing or publicly available either before the date of an invention or more than one year prior to the patent application date [51].

USPTO United States Patent and Trademark Offce

CHAPTER I

INTRODUCTION

Software has traditionally been under the realm of copyright and trade secret law, but it has entered a new realm over the past thirty years: patent law. For many years, the United States Patent and Trademark Offce did not allow software to be patented, but since 1981, when the United States Supreme Court ruled in the *Diamond, Commissioner of Patents and Trademarks v. Diehr* case, the United States Patent and Trademark Offce has allowed software to be patented. Universities then began protecting their software inventions with patents so that the software inventions could be licensed to commercial companies.

The licensing of university inventions has led to an infusion of money into research programs. This infusion of money drives more research, which in turn leads to more licenses. The protection of the university's inventions is paramount in this infusion process. Companies will not license an invention from a university and spend millions of dollars to commercialize an invention and then have a competitor come along and copy the invention [37]. Companies are important to universities because without commercialization most universities could not fund the cost of obtaining patents. The validity of a patent is important to any commercial venture that utilizes the protection offered by the patent. However, the inventor never knows how strong the patent claims are until the product asso-

ciated with the patent is challenged in court. For these reasons, it is important for inventors to understand the patent process at a university.

The risks associated with university inventions are as follows:

1. Not fling a patent application
2. Patent application not being allowed
3. Patent not holding up in court

These risks are related to the hypothesis and research questions of this thesis. Inventors at a university can lower the occurrence of the above listed risks by understanding and applying the information contained in this thesis.

1.1 Hypothesis

The hypothesis of this research is that:

> An inventor at a university can improve the protection of his or her software patent by applying certain information about patent prosecution practices and the relevant prior art.

1.2 Questions

The following are the research questions designed to provide evidence for or against the hypothesis.

1. How does the knowledge of prior art help inventors at a university?
2. How does the understanding of patent laws and regulations help inventors at a university?
3. What are the differences between a university and corporate environments from an inventor's viewpoint?

1.3 Relevance

Inventors at a university go through a standard process when disclosing an invention to university technology licensing offces. The basic steps are as follows [42]:

1. Maintain complete laboratory fles and notebooks
2. File before publication or presentation
3. Submit an invention disclosure form
4. Review by technology licensing offce
5. File provisional patent application
6. Market and develop
7. Prepare patent application
8. Prosecute patent application

The process of protecting inventions at a university could be accelerated by inventors who understand the principles behind these steps. Most of the steps of protecting university inventions are directly related to the basics of patent law and regulations along with how the university environment operates. These steps are thus directly related to the research questions presented in this thesis.

CHAPTER II

PATENT HISTORY

2.1 U.S. System

This chapter examines the U.S. patent system to determine what university software inventors can do to ensure that they have the proper protection for their inventions. Some of the concerns that face these inventors are as follows:

1. Proper laboratory notebooks and documentation [32]
2. Using diagrams such as functional block diagrams, fo wchart diagrams, and state diagrams [48]
3. Writing invention disclosures that fully describe the invention [3]
4. Understanding the prior art [63]
5. Communicating the scope of their invention to the patent practitioners that are writing the patent application [57]

The challenges of patenting software are quite different than copyright procedures. The knowledge of prior art ensures that the inventor is not "reinventing the wheel," not infringing on a patent, and understands the current state of the art [63]. The knowledge of patent laws, diagrams, readability, and prior art enables a software inventor to take control of the protection of his or her invention to ensure that the application of this information leads to improvement in the application process [57]. Kirsch and Skulikaris discuss the importance "for software developers to have at least an elementary knowledge of IP [intellectual

property] protection" [35]. The inventors have to be concerned with patent practitioners who do not completely understand their invention and should carefully analyze the patent application for misunderstandings regarding the invention [57].

2.2 University Inventors

University technology licensing offces help inventors in many ways [42]. One such beneft occurs when an inventor has an invention that could possibly have great commercial potential. Most university technology licensing offces work with the inventor to protect the invention, market the invention to companies in the given feld, and fnally , license the invention to the company willing to give royalties to the university [42]. The advantage to the inventor is that at most universities, the inventor and the inventor's department receive a percentage of the royalties [41]. Not only does this method bring in money for the inventor, but it increases the inventor's chances for job advancement in the department.

2.3 Copyright versus Patent

Most companies have shifted their software protection towards patent protection because of the limitations of copyright protection [34]. Some of the limitations of copyrights were revealed in the *Apple v. Microsoft* case [21]. In the *Apple v. Microsoft* case, some of the interface modules were not protected under copyright [21]. Copyright protects the expression of an idea, while a patent protects the idea itself [52]. One problem with copyright is that someone can write a new program that is "exactly or substantially the same

as the copyrighted program, and the lack of copying and access to the original program negates infringement of the copyright" [44]. Although patent protection costs are signif - cantly higher than copyright protection, patent protection is much broader and is most of the time considered worth the investment [33].

2.4 History of Software Patents

Before *Diamond, Commissioner of Patents and Trademarks v. Diehr*, there were several cases involving software patents in which the Supreme Court ruled that software was unpatentable [40]. One such case is *Parker v. Flook*, which dealt with "a method for updating an 'alarm limit' in the catalytic conversion of hydrocarbons" [40]. The Supreme Court thought that the claims in this patent application were strictly tied to scientifc principles and thus were not patentable [40].

The trend of not allowing computer-related patents continued in the Supreme Court until *Diamond, Commissioner of Patents and Trademarks v. Diehr* in 1981 [40]. This case involved a method for curing rubber and the Supreme Court ruled that the involvement of a computer did not automatically mean that the method could not be patented [40]. This court case has opened the door to thousands of software patents in the years since its ruling. Currently, software patents are described in broad terms without the description of the actual computer system in detail [11]. However, some software patents still describe the entire computer with input and output devices as the frst fgure of the patent [10], such as the example in Figure 2.1.

FIG. 1

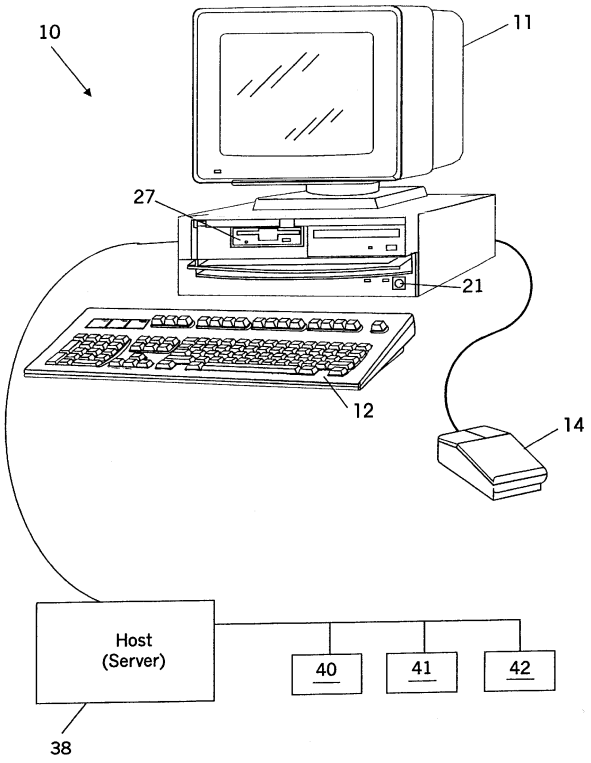

Figure 2.1 Cheston et al. Figure 1

CHAPTER III

RELATED WORK

3.1 Cross-Case Analysis

Seaman discusses “the human role in software development” [54]. Studying this human role produces qualitative data which “can be adapted and incorporated into the designs of empirical studies in software engineering” [54]. Cross-case analysis can be used to “build up the weight of evidence in support of a” hypothesis [54]. Cross-case analysis involves the following steps [54]:

1. Dividing the data
2. Identifying the relevant variables
3. Identifying the relationships between the variables

Cross-case analysis has the beneft of allowing data from different contexts to be compared to each other to provide support for a research question or hypothesis [54].

3.2 Software Patentability

Software is like any other method except that it is implemented on a computer system [49]. This quality does not make software unpatentable; it simply ensures that it receives more scrutiny from the United States Patent and Trademark Offce before allowance [60].

The USPTO has special guidelines for computer-related inventions to ensure that non-patentable material is not allowed [60]. The important aspect is that the software invention must be “useful, concrete, and tangible” [40].

Nixon and Davidson discuss the concept of a software program being patented [49]. While it is important to understand that software can do novel processes just like hardware. It is easy for people to understand what a new physical device does, it is often diffcult to understand the process that is occurring in the software. Software developers need to understand that after “semantic misunderstandings are stripped away, programs are just as potentially patentable subject matter as anything else under the sun made by humans” [49].

Durham cites the diffculty “in two long-standing doctrines” as the reason software patents have faced a challenge in being allowed [15]. The doctrines are the unpatentability of mathematical algorithms and the mental steps doctrine. The mathematical doctrine is that “the truths of mathematics are considered beyond the possibility of ownership” [15]. The mental steps doctrine states that “a process is unpatentable if an essential step of the process requires human thought” [15]. The mental steps doctrine was used for many years to as one of the reasons why software should not be patented [15]. The mental steps doctrine is longer used as a valid argument against software patents [15].

There are many challenges that surround allowing software patents, some of which are discussed in the following works. The Committee on Science of the U.S. House of Representives sponsored a hearing on the patent system that discusses the challenges that

software patents pose to the patent examiners because of the increased number of patent applications [12]. Blakemore discusses the differences between U.S. and European patent law in regard to software [4].

Although software patents are now allowed, there are numerous works that discuss why software should not be patented. Some of the reasons why software patents should not be allowed are:

- Preparation and prosecution takes a long time [19]
- Too much prior art exists [19]
- Easy to code around patents [46]
- No requirement to conduct prior art search [50]

Boyle discusses the group of software developers that do not think that software should be patented [5]. Nichols discusses some of the reasons why software should not be allowed to be patented [46]. Garfnk el, Stallman, and Kapor discuss the reasons that patents on software should not be allowed and that "patents can't protect or invigorate the computer software industry; they can only cripple it" [19]. Davis et al. discuss that "most software is innovative rather than inventive" and thus not patentable [14]. Harris discusses some of the reasons why software patents are impractical [20]. Stern discusses some of the problems associated with allowing software patents that could impede competition [55]. O'Reilly discusses the problems with allowing patents without a prior art search requirement and thus "all is not well in the Internet and e-commerce industry" [50]. Stern discusses the diffculty involved in deciding how to limit the types of software patents that should be allowed [56].

3.3 Disclosure

Stobbs discusses the need to fully disclose a software invention in the specifcation to ensure compliance with Title 35, United States Code, Section 112, frst paragraph [57] (abbreviated as 35 U.S.C. §112). The specifcation must include the following [60]:

1. A written description of the invention
2. The manner and process of making and using the invention ("enablement requirement")
3. The best mode contemplated by the inventor of carrying out his invention

Stobbs explains the need for the three requirements in 35 U.S.C. §112, frst paragraph, and why it is important that a software patent application's specifcation meets these requirements [57]. Patent examiners have a limited amount of time to analyze applications. Therefore it is important that applications abide by these guidelines. Patent examiners are not the only people who read patents. Business people read patents to decide whether to invest time and money on an invention. Judges and juries read patents to determine the outcome of court cases. There are numerous audiences that patent practitioners have to consider when they are composing and prosecuting a software patent application [57].

Natoli discusses the use of providing the source code as "a substitute for dozens of fo wcharts of minor but important subroutines of the software" [44]. This inclusion of source code in a software patent is not required, but the inclusion does help meet the requirements in 35 U.S.C. §112. There are a few concerns with including the source code in a patent such as the providing of source code to competitors, but the benefts of providing the source code sometimes outweigh the possible risks [44].

Nigon discusses the importance of the written description in a patent [48]. Nigon explains that a patent practitioner has a duty to fully explain every detail of the invention even if the inventor "believes that the functioning of a particular process is well known and need not be described in detail" [48]. Inventors overestimate what is well known in the feld and sometimes believe that simple concepts should not be explained, but this behavior can lead to a failure to fulfll the requirements of 35 U.S.C. §112. Burge thinks that a patent application "should be a readable and understandable teaching document" and "set forth the pith of the invention in terms a grade-school student can grasp" [9]. It is therefore important that the inventor understand what is required to make a well-drafted patent application.

Balconi-Lamica explains the importance of invention disclosures and the legal ramifcations of not fully describing the invention in the disclosure [3]. Most of the time, an inventor writes an invention disclosure for the employer, whether a company or university. An invention disclosure should fully describe the problems solved, the features, the advantages over prior art, and the implementation of the invention. A well-written invention disclosure will not only help a patent practitioner write a patent application, but it will also establish a date of conception. Balconi-Lamica believes that "writing an invention disclosure is a learned skill" [3]. Thus, an inventor should understand the parts of a disclosure before he or she begins writing it.

3.4 Written Description

Holmes explains that “experts in law, not technology, make the legal judgments” [25]. The exact wording of patent claims is important because that is what the legal judgments are based on. The wording must come from a commonly accepted source, although sometimes the wording is interrupted in a non-common way because of the precedents in the court system [25]. The knowledge of these precedents makes it important that inventors and patent practitioners understand the importance of the words in a patent claim [25].

Nigon explains the value of using diagrams to describe a software invention [48]. The three types of diagrams that aid in the enablement of a patent are functional block, fo wchart, and state. Functional block diagrams are useful to “show essential connections among the processes and links between each of the processes and the relevant data structures” [48]. Flowchart diagrams are useful to show the sequence of steps implemented in a software invention. State diagrams are essential when describing the timing of the processing in a software invention. These diagrams are aids to ensure that a patent’s specifcation fully describes and discloses the invention.

3.5 Prior Art

Wiens explains the importance of searching prior art “before, during, and after the development of an invention” [63]. Wiens also thinks it is important to “perform more exhaustive searches to answer questions regarding validity or infringement of patents”

[63]. Knowing the current state of the feld allows innovators to not only avoid infringing on a competitor's patent, but it also helps innovators fnd solutions to diffcult problems.

Marcus explains the benefts of mining patent information for various uses [38]. A few such uses are [38]:

- Prevent duplication of research & development
- Identify experts in a specifc feld
- Find solutions to technical problems
- Generate ideas
- Establish state of the art

3.6 Ownership

Neitzkel discusses the ownership of inventions in regards to the employee relationship to the employer [45]. It is important to realize who owns what is in the employee's head. There are many issues that have to be raised when an employee invents something, and those issues should be addressed before they become a problem. In the university setting there are many issues regarding sponsorship that have to be addressed before ownership can be decided [41]. One such issue is that if federal money is used to sponsor research, the federal government will have a royalty-free license to use any invention that stems from that research [41]. Neitzkel thinks the ownership depends on many factors such as [45]:

- Nature of the invention
- Date of conception

- Duties of the employee

Ku discusses the approach Stanford takes toward the ownership of software inventions [36]. This approach pertains to determining "if software has been developed with more than incidental use of university facilities" to decide whether Stanford should have rights to the software invention [36]. Ku also discusses some success stories of software that Stanford has licensed, such as the Google search engine, a discrete Fourier transform, and MINOS [36]. Ku explains that software the university retains ownership of can be successfully licensed and be a commercial success in many different ways [36].

3.7 Infringement

Another concern for software developers is infringing on the numerous software patents that have already issued. Nichols discusses this risk and thinks that because of the cost of searching "every algorithm, interface, and data structure in a product" most software developers will not do prior art searches [47]. Nichols suggests several steps to minimize the risk of infringing a patent [47]:

- Document the sources of algorithms
- Documentation disclosure
- Source code availability

Burge explains the importance of inventors researching the feld to ensure that they "don't reinvent the wheel" [9]. Patents can also be used as a technical resource for a savvy inventor or engineer. Burge stresses the importance of a patentability search to ensure

that the proper scope of patent protection can be achieved by a patent application. A patentability search can sometimes save the inventor time and money by determining that the invention cannot be patented.

3.8 Patent Portfolios

Bragg discusses the shift from copyright protection of software to patent protection [6]. Bragg also discusses the general characteristics of most of the other related work in this chapter, such as the benefts of having a large patent portfolio, the problems with not completing a prior art search, and the patenting of common programming techniques. The challenge of programming is that "it is almost certain that some idea in your code that you believe to be nonunique, obvious, or covered by prior art is in fact protected by some patent or described in some pending application" [6].

Kahin explains the problem of large software packages that expose software developers to legal action [31]. Software packages "contain thousands of separately patentable processes, each of which adds to the risk of infringing patents that are already in the pipeline" [31]. This concern leads to the fear that small software developers will not be able to compete with the large software developers because of the lack of a patent portfolio [31], since most large companies use their patent portfolio as a bargaining platform in infringement lawsuits [31]. The large companies trade rights to use certain patents in their portfolio with other companies [31]. Thus, small companies have a diffcult time avoiding infringement lawsuits compared to their larger counterparts [31].

Galler discusses the use of patents as bargaining tools between companies [18]. Software companies "are now applying for patents left and right, in order to have some of their own with which to bargain and/or trade" [18]. The software companies are worried that they will be involved in a lawsuit and will not have a large enough portfolio.

Aharonian discusses the quality of software patents between large and small companies [1]. Large companies want a large patent portfolio and small companies want high-quality patents. Inventors and patent practitioners are not required to do any searching "to prove their inventions are novel and not obvious" and therefore most do not, which leads to low-quality patents [1]. Low-quality patents are associated with the third risk described in Chapter I. This risk is that the patent will not hold up in a court of law. This risk of a patent not holding up stems from patents that are mass produced for the purpose of flling a patent portfolio. Low-quality patents are patents that are allowed on marginal advances in the technology and are only deemed suitable for use in a larger patent portfolio. Low-quality patents are thus the product of companies who want "to extract hundreds of millions of dollars in royalties and control the marketplace" [1]. All software developers need to be concerned about low-quality patents and work to encourage the United States Patent and Trademark Offce, all companies, and patent practitioners to modify the system to weed out the low-quality patent applications [1].

There are numerous other works that discuss some of the characteristics and alternatives of patent portfolios. Duvall discusses the large number of software patent applications that companies are submitting to protect themselves from patent infringement

lawsuits [16]. Wolfe discusses the large companies that are “corralling huge numbers of software patents“ to protect themselves from patent infringement lawsuits [64]. Wade discusses the role patents play in the protection of intellectual property in a research department [62]. Duvall and Judge explain that companies want a large number of patents in their portfolio so that when a competitor claims that they are infringing one of its patents, they can cross-license one or more of their patents to satisfy the competitor and not be brought to court [16, 64]. Judge discusses how Netscape “posted on its Web site an appeal for evidence of prior art” when they were being sued by Wang Global for patent infringement, thus using the Internet community as a replacement for a patent portfolio [30]. Wang lost its case against Netscape because of “the outpouring from software developers” of prior art against Wang. This case has shown that there is an alternative to having a patent portfolio [30].

3.9 University Environment

There are several works on how universities operate and how the corporate environment is different from universities. Frank discusses some of the reasons that universities do not license as many of their inventions as could be licensed [17]. Some of the reasons why universities have diffculties licensing their inventions include:

- Negotiations get slowed down by the university system
- Companies has little control over research priorities
- Technology is not market-ready
- Goals of licensing are not clear

- Lines of authority are not well established
- Inconsistent priorities

Henry et al. discuss the relationship between the number of invention disclosures received and the number of patent applications fled in the university and commercial environments [24]. Some of the issues surrounding this relationship along with the university and commercial stance on the issue are shown in Table 3.1 [24].

Table 3.1 Differences between the University and Commercial Environment

Issue	University	Corporate
Patent strategy	Explore market before fling	Build patent portfolio
Defensive patents	No need	Block competitors
Inventions overall	Small fraction patented; Most freely available upon publication	Nearly all commercial inventions are patented

CHAPTER IV

CASE STUDIES

4.1 Criteria for Selecting Cases

The case studies were selected to obtain a wide selection of patents according to the items in Table 4.1. These criteria were used to ensure that a signifcant variety of patents were selected for case studies. The date of patent was chosen to illustrate the development of software patents over the past twenty years. The classifcation of the patent was chosen to ensure that a sampling of patents from different felds was used. The USPTO uses patent classifcation to categorize patents into different felds and subfelds for reference and organizational purposes. The number of drawings were used to ensure that the case studies had a signifcant number of fgures for reference in the analysis phase of research. The number of claims were used to illustrate the wide variety of the types of claims used in software patents. The number of references were used to illustrate the amount of prior art searching that was completed for each case study. These criteria were selected and used to ensure enough examples could be found to support the research. Table 4.2 shows a listing of the criteria information for each case study.

To fulfll the criteria, the case studies were selected by a few different methods. The case studies were initially selected from a list of famous software patents [29]. The following case studies were selected from that list:

- Heizer
- Inoue
- Brown
- Auslander et al.
- Hellman et al.

Some of the case studies were selected from an IBM website listing its important patents [28]. The following case studies were selected from that website:

- Ciacelli et al.
- Cheston et al.

The remaining case studies were selected using the criteria in Table 4.1 to fll in the gaps of classifcation and year coverage. The remaining case studies selected using this method follow:

- Moudgal et al.
- Teng et al.
- Bullwinkel et al.

Although the case studies were selected from several different sources, the end result was a group of case studies that fulflled the goals of this research.

Table 4.1 Criteria Used for Selection

Criteria
Date of patent
Classifcation of patent
Number of drawings
Number of claims
Number of references citied

Table 4.2 Criteria for Each Case Study

U.S. Patent	Date Issued	Classifcation	Drawings	Claims	References
Moudgal et al.	Feb. 12, 2002	711/133	12	8	11
Teng et al.	Dec. 4, 2001	358/1.15	14	13	42
Ciacelli et al.	May 22, 2001	380/5	4	41	17
Cheston et al.	Feb. 27, 2001	709/221	3	14	10
Bullwinkel et al.	Aug. 8, 2000	434/118	22	7	4
Heizer	Sep. 28, 1993	395/650	5	20	3
Inoue	Dec. 29, 1992	395/800	15	12	21
Brown	Oct. 23, 1990	364/900	2	3	3
Auslander et al.	Apr. 7, 1987	364/300	3	8	1
Hellman et al.	Apr. 29, 1980	178/22	6	8	2

4.2 Case Study: Moudgal et al.

U.S. Patent 6,347,360 protects an ”apparatus and method for protecting cache data from eviction during an atomic operation” [43]. The eleven prior art references were all cited by the patent examiner as shown in Figure 4.1. Thus, the inventors and patent practitioner(s) did not do a prior art search before fling the patent application.

This patent was allowed on the frst offce action and therefore, the prior art must not have conficted with the patent claims as fled. Prior art is the state of knowledge in the feld and not just the knowledge of a specifc invention.

4.3 Case Study: Teng et al.

U.S. Patent 6,327,045 protects “an implementation of a computer network which provides the ability for a network client to submit data to a network server for performing a job at a logical endpoint associated with the network server” [58]. An extensive number of prior art references were cited by the inventors and the patent practitioner. There is a total of forty-two prior art references cited in this patent. The examiner cited only seven of the forty-two references and the applicant cited nine non-patent prior art references. The applicant obviously completed a prior art search before submitting the patent application to the United States Patent and Trademark Offce. The prior art references are shown in Figure 4.2 and Figure 4.3.

US006347360B1

(12) **United States Patent**
Moudgal et al.

(10) **Patent No.: US 6,347,360 B1**
(45) **Date of Patent: Feb. 12, 2002**

(54) **APPARATUS AND METHOD FOR PREVENTING CACHE DATA EVICTION DURING AN ATOMIC OPERATION**

(75) Inventors: **Anuradha N. Moudgal**, Fremont, CA (US); **Belliappa M. Kuttanna**, Austin, TX (US); **Allan Tzeng**, San Jose, CA (US)

(73) Assignee: **Sun Microsystems, Inc.**, Palo Alto, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/513,033**

(22) Filed: **Feb. 25, 2000**

(51) **Int. Cl.**[7] **G06F 12/00**

(52) **U.S. Cl.** **711/133**; 711/155; 711/156; 711/144; 711/145; 711/140; 711/146

(58) **Field of Search** 711/133, 135, 711/140, 144, 145, 146, 154, 155, 156, 143; 709/104; 710/200, 52, 56, 48, 260, 262, 263; 712/244

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,428,761	A *	6/1995	Herlihy et al.	711/130
5,617,556	A *	4/1997	Baumgartner et al.	711/141
5,627,992	A *	5/1997	Baror	711/133
5,706,464	A *	1/1998	Moore et al.	711/122
5,905,998	A *	5/1999	Ebrahim et al.	711/144
6,038,645	A *	3/2000	Nanda et al.	711/141
6,049,851	A *	4/2000	Bryg et al.	711/141
6,098,156	A *	8/2000	Lenk	711/146
6,212,601	B1 *	4/2001	Shiell	711/118
6,212,602	B1 *	4/2001	Wicki et al.	711/122
6,212,605	B1 *	4/2001	Arimilli et al.	711/133

* cited by examiner

Primary Examiner—Hong Kim
(74) *Attorney, Agent, or Firm*—Pennie & Edmonds LLP

(57) **ABSTRACT**

Apparatus and method for protecting cache data from eviction during an atomic operation. The apparatus includes a first request queue, a second request queue, and an atomic address block. The first request queue stores an entry for each cache access request. Each entry includes a first set of address bits and an atomic bit. The first set of address bits represents a first cache address associated with the cache access request and the atomic bit indicates whether the cache access request is associated with the atomic operation. The second request queue stores an entry for each cache eviction request. Each entry of the second request queue includes a second set of address bits indicating a second cache address associated with the cache eviction request. The atomic address block prevents eviction of a third cache address during the atomic operation on the third cache address. During a first clock cycle the atomic address block receives and analyzes a first set of signals representing a first entry of the first request queue to determine whether they represent the atomic operation. If so, the atomic address block sets a third set of address bits to a value representative of the first cache address. During a second clock cycle during which the atomic operation is being executed the atomic address block receives and analyzes a second set of signals representing the second set of address bits to determine whether the second set of address bits represent a same cache address as the third set of address bits. If so, the atomic address block stalls servicing of the second request queue, thus preventing eviction of data from the cache upon which an atomic operation is being performed.

8 Claims, 8 Drawing Sheets

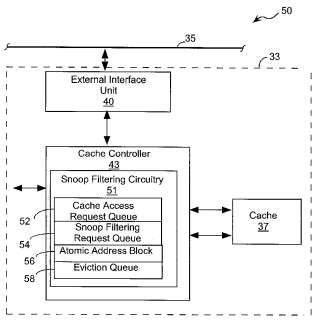

Figure 4.1 Moudgal et al.: Page 1

US006327045B1

(12) **United States Patent**
Teng et al.

(10) **Patent No.: US 6,327,045 B1**
(45) **Date of Patent: *Dec. 4, 2001**

(54) **COMPUTER NETWORK**

(75) Inventors: **Chia-Chi Teng**; **Babak Jahromi**, both of Redmond, WA (US)

(73) Assignee: **Microsoft Corporation**, Remond, WA (US)

(*) Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **08/932,772**

(22) Filed: **Sep. 18, 1997**

(51) **Int. Cl.**[7] **G06F 15/00**

(52) **U.S. Cl.** **358/1.15**; 395/831

(58) **Field of Search** 395/114, 200.03, 395/200.33, 200.47, 200.49, 200.57, 200.59, 200.36, 200; 358/1.15, 200; 709/217–219, 203, 206, 213

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,537,626	7/1996	Kraslavsky et al.	395/828
5,572,643	11/1996	Judson	395/793
5,625,781 *	4/1997	Cline et al.	395/335
5,687,320	11/1997	Wiley et al.	395/200.16
5,708,780	1/1998	Levergood et al.	395/200.12
5,715,453 *	2/1998	Stewart	395/615
5,721,908	2/1998	Lagarde	395/610
5,732,219	3/1998	Blumer et al.	395/200.57
5,734,831	3/1998	Sanders	395/200.53
5,740,231	4/1998	Cohn et al.	379/89
5,742,768	4/1998	Gennaro et al.	295/200.33
5,742,845	4/1998	Wagner	395/831
5,745,360	4/1998	Leone	364/140
5,752,246	5/1998	Rogers et al.	707/10
5,754,830	5/1998	Butts et al.	395/500
5,764,908	6/1998	Shoji et al.	395/200.47
5,765,176 *	6/1998	Bloomberg	707/614
5,774,670	6/1998	Montulli	395/200.57

(List continued on next page.)

OTHER PUBLICATIONS

IBM* Technical Disclosure Bulletin in vol. 38 No. 12, Dec. 1995.*

Pipline's Internet Printing system lets the printer surf the Web (the OBSERVER, pp. 45–46), Mar. 1997.*

"Printer MIB",. Mar. 1995, Smith et. al., Internet Network Working Group.

"Internet Printing Protocol/1.0", Jul. 25, 1997 DeBry et al., Internet Network Working Group.

"Inside Windows NT", Custer, 1993, Chapter Nine entitled "Networking".

"Hypertext Transfer Protocol–HTTP/1.1", Jan. 1997, Fielding et al., Internet Network Working Group.

"An Extension to HTTP: Digest Access Authentication", Jan. 1997, Franks et al., Internet Network Working Group.

180/IEC IED175–2; Sep. 1, 1996.

180/IEC 10175–1; Sep. 1, 1996.

Primary Examiner—Joseph Mancuso
Assistant Examiner—Douglas Tran

(57) **ABSTRACT**

An implementation of a computer network which provides the ability for a network client to submit data to a network server for performing a job at a logical endpoint associated with the network server. By way of example, the data may be a print job, the job a printing operation, and the logical endpoint a printer. The logical endpoint is assigned a uniform resource locator (URL) address which allows the data to be routed thereto and the computer network communication messages are formatted into the hypertext transfer protocol (HTTP). A further implementation of the computer network also provides the network client with the ability to perform system administration utilizing a standard Internet browser application.

13 Claims, 14 Drawing Sheets

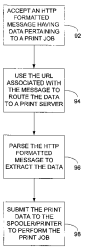

Figure 4.2 Teng et al.: Page 1

U.S. PATENT DOCUMENTS

5,787,254	7/1998	Maddalozzo et al.	395/200.58
5,790,790	8/1998	Smith et al.	395/200.36
5,790,793	8/1998	Higley	395/200.48
5,793,966	8/1998	Amstein et al.	395/200.33
5,796,393	8/1998	MacNaughton et al.	345/329
5,799,206	8/1998	Kitagawa	395/856
5,802,530	9/1998	Van Hoff	707/513
5,826,242	10/1998	Montulli	705/27
5,867,636 *	2/1999	Walker	395/114
5,898,835 *	4/1999	Truong	395/200.47
5,901,286	5/1999	Danknick et al.	395/200.33
5,905,908 *	5/1999	Wagner	395/831
5,946,458 *	8/1999	Austin et al.	395/114
5,974,234 *	10/1999	Levine et al.	395/114
5,987,480	11/1999	Donohue et al.	707/501

* cited by examiner

Figure 4.3 Teng et al.: Page 2

4.4 Case Study: Ciacelli et al.

U.S. Patent 6,236,727 protects ”an apparatus, method and computer program product for processing a data stream scrambled” [11]. This patent has a fo wchart diagram that Nigon advocated [48]. Figure 2 of U.S. Patent 6,236,727 is shown in Figure 4.4. International Business Machines Corporation (IBM) fled this patent on June 24, 1997 before encrypted data for copyright protection was widely used in commercial products which indicates its use in IBM’s patent portfolio.

4.5 Case Study: Cheston et al.

U.S. Patent 6,195,695 protects “a system and method for recovering from corruption of an executable application and/or operating system stored on a client computer without downloading another copy of the application and/or operating system” [10]. The ten prior art references were all cited by the patent examiner, as shown in Figure 4.5. Therefore, the inventors and patent practitioner(s) did not do a prior art search before fling the patent application. Although this patent was not allowed as the frst offce action from the United States Patent and Trademark Offce, it was allowed on the second offce action.

The frst fgure of this patent shows the use of the personal computer in the context of the embodiments of the patent. The frst fgure is shown in Figure 2.1. The brief and detailed description of the frst fgure is shown in Figure 4.6. These descriptions show the physical means that are used in this patent.

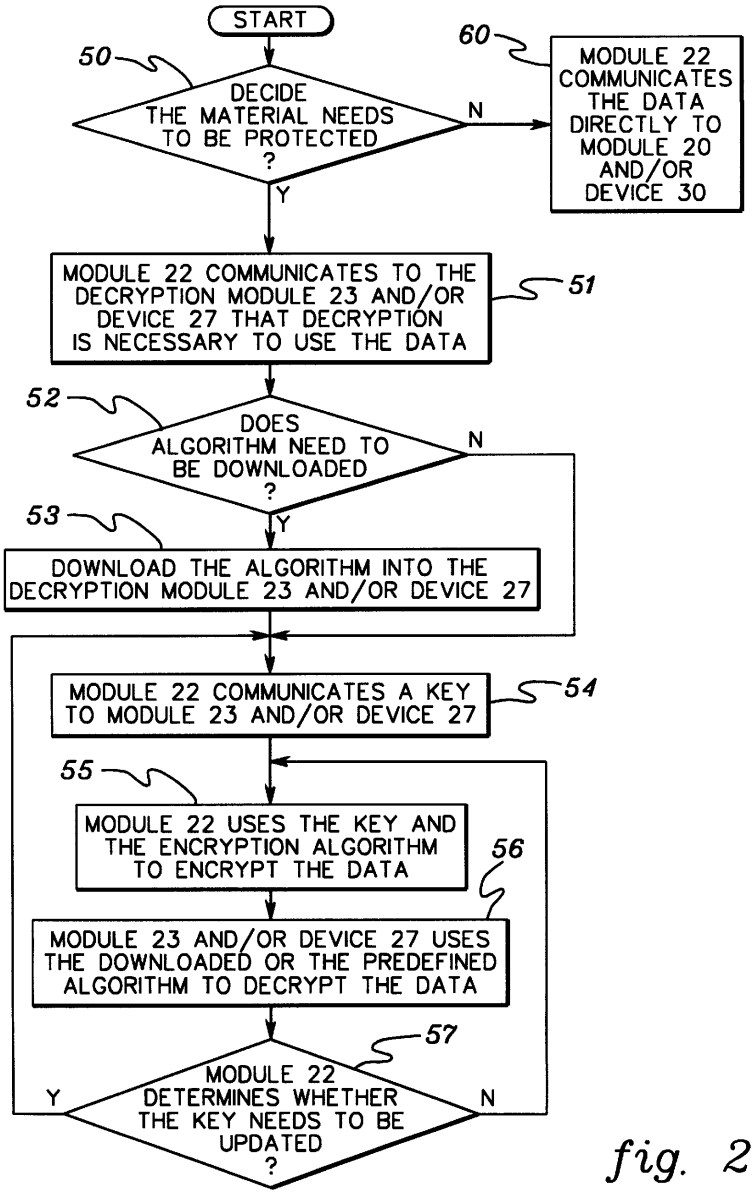

fig. 2

Figure 4.4 Ciacelli et al.: Figure 2

US006195695B1

(12) **United States Patent**
Cheston et al.

(10) **Patent No.: US 6,195,695 B1**
(45) **Date of Patent: Feb. 27, 2001**

(54) **DATA PROCESSING SYSTEM AND METHOD FOR RECOVERING FROM SYSTEM CRASHES**

(75) Inventors: **Richard W. Cheston**, Morrisville; **Howard Locker**, Cary; **David B. Rhoades**, Apex, all of NC (US)

(73) Assignee: **International Business Machines Corporation**

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/181,138**

(22) Filed: **Oct. 27, 1998**

(51) **Int. Cl.**[7] **G06F 15/177**

(52) **U.S. Cl.** **709/221**; 709/201; 709/203; 709/212; 709/217; 709/248; 707/200; 707/202; 707/204; 711/161; 711/162; 714/2; 714/6

(58) **Field of Search** 709/201–203, 709/212–213, 217–222; 711/248, 100, 161–162; 713/200–201; 707/200–205, 10; 714/1–8

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,375,232	*	12/1994	Legvold et al.	707/200
5,375,240	*	12/1994	Grundy	713/200
5,381,545	*	1/1995	Baker et al.	707/202
5,390,335	*	2/1995	Stephan et al.	709/221
5,745,905	*	4/1998	Larsson et al.	707/203
5,802,297	*	9/1998	Engquist	709/212
5,845,082	*	12/1998	Marakami	709/226
5,857,208	*	1/1999	Ofek	707/204
6,035,412	*	3/2000	Tamer et al.	707/204
6,101,497	*	8/2000	Ofek	707/10

* cited by examiner

Primary Examiner—Ahmad F. Matar
Assistant Examiner—Bharat Barot
(74) *Attorney, Agent, or Firm*—George E. Grosser

(57) **ABSTRACT**

A system and method for recovering from corruption of an executable application and/or operating system stored on a client computer without downloading another copy of the application and/or operating system. This recovery is accomplished by storing, preferably at the initial program loading, two copies of the application (and the operating system, if desired) in different segments of a partitioned storage, one copy as a working copy and a second copy as an archive or backup copy. When the working copy of the application or operating system becomes corrupted and crashes, the backup copy is thereafter used as the new working copy and, if desired, a new backup copy is stored to be used when the working copy crashes. This allows the system to continue functioning after a crash without the necessity to find a new copy of the application and operating system from outside the computer.

14 Claims, 3 Drawing Sheets

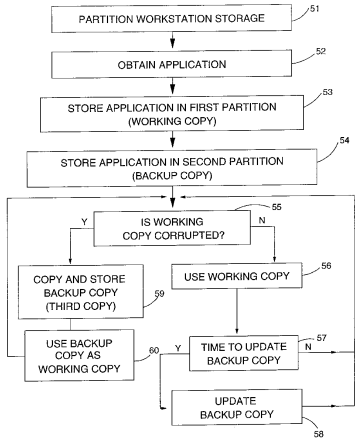

Figure 4.5 Cheston et al.: Page 1

to have a backup of the stored data). In the case that the updating is from the server downloading a copy of the image, Such updating may be scheduled so as to occur when the data traffic on the communications channel is at a low level, for example, in the middle of the night, or may be available for those times when the traffic is at a reduced level. Thus, for example, a copy of the executable application could be downloaded once a week at 4:00 a.m. on Sunday morning. Alternatively, the system could be programmed to back itself up every evening or once a week, as desired, and then, if the system crashes, the content of the backup copy is no more out-of-date than the time since the last backup. In some applications, it may be desirable to retain backup copies of different period, so you might backup the application with one copy from this past weekend and a second copy from the previous weekend, and then when each backup occurs, to keep the most recent two versions.

Accordingly, the present invention overcomes the disadvantages and limitations of the prior art by providing a workstation system which allows recovery of an executable application and/or operating system after the executable application and/or operating system has been corrupted. This recovery is effected without having to download a replacement copy and without requiring a copy of the application and the operating system in removable media at the workstation.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a view of a computer network of the type which is useful for using the present invention.

FIG. **2** is a logical flow chart showing the operation of the present invention allow a workstation to recover from system crashes.

FIG. **3** is a schematic view of the memory of a personal computer in the computer network of FIG. **1**, using the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

FIG. **1** shows a workstation or personal computer **10** operating as a client computer in a client/server data processing environment. The personal computer **10** is shown with a monitor **11**, a keyboard **12**, a mouse **14**, a floppy drive **27** and a power switch **21**. The personal computer **10** also includes within the case a hard drive or storage device (not shown) for storing executable applications (programs), operating system and/or data. This hard drive is of conventional design and is commercially available from any one of a variety of manufacturers and usually would include the capability to partition the storage into a plurality of segments in a conventional and well-known manner. Many hard drives also have indicators to indicate which partitions or areas of the storage are "active" or available for use, and which are "hidden" or not currently available for use.

The personal computer **10** is connected to a host computer or server computer **38** through a communication channel **36**. The communications channel would also connect the host computer or server **38** to a plurality of other client computers, shown here by blocks **40**, **41**, **42**. The number of client computers depends on the application and the data processing involved, but could include a large number of clients operatively connected to the host computer **38** through the communications channel **36**. The communications channel **36** used in the present invention could be either a local-area-network (LAN), a wide-area-network (WAN) or a radio frequency (RF) communications system of any of a variety of known and conventional designs. In any case, the communications channel may be the limiting factor in how much data is communicated from the client computers to the host and, if the communications channel were also used to bring back up client computers after the images have been corrupted, that additional communications channel traffic could limit the amount of data being sent from the client computers to the host computer.

FIG. **2** illustrates the logical flow of the present invention in flowchart form. The first step in this process is that the storage (the hard drive discussed with respect to FIG. **1**, in its preferred embodiment) is partitioned to be divided into a plurality of different segments at block **51**. The executable application (and operating system, if desired) is then obtained at block **52**, either by downloading the application from the host or loading it from removable media.

At block **53**, a first copy (also referred to as a "working" copy) of the executable application (and, optionally, the operating system) is stored in a first partition and at block **54** a second copy (sometimes referred to as an archive or backup copy) of the executable application and the operating system is stored into a second partition of the storage. At block **55** is a test as to whether the working copy of the application and operating system is corrupted: this test is simple, since current versions of the operating system typically simply "hang" or "crash" and require that the user turn the entire workstation off, then restart the machine, but designs could easily evolve to allow the application and/or operating system to test themselves as to whether either is "corrupted" or not. In any event, if the working copy is not corrupted, then the working copy of the executable application (and operating system) is used, see block **56**, for the data processing (i.e. the application might be order entry or word processing or other application, as desired). In block **57**, the system tests whether it is time to update the backup copy, a updating which may occur after a predetermined amount of time has passed or after a predetermined amount of data has been accumulated, or because it is a predetermined time (like 4:00 a.m. on Sunday morning when the system usage might be expected to be low, allowing the use of the communications channel for updating). If the backup copy of the application is to be updated, it is updated at block **58**, then returns to block **55** to resume the cycle.

If the working copy of the executable application (and operating system) has been determined to be corrupted at the block **55**, then at block **59** the backup copy is copied (a third copy), the backup copy becomes the working copy and the third copy becomes the backup copy at block **60**, and the data processing resumes at the block **55**.

FIG. **3** illustrates the organization of the workstation storage **100** (the hard drive of the personal computer) as used in practicing the present invention. The storage **100** is partitioned or divided into segments or partitions in a conventional manner, and shown here with four segments or partitions, **101**, **102**, **103** and **104**. When the personal computer is turned on, or boots up, the program always goes to "image **0**" segment which is reference number **101**. Stored within this image **0** segment is an address of the "working" copy of the application (and operating system, if desired), in this case "address **1**", in location **101***a*. As the executable application is loaded, a copy is put into segment **102** and segment **103** as image **1** and image **2**, respectively, located at address **1** and address **2**, respectively. Each of the segments **101**, **102**, **103** also includes an indicator **102***b*, **103***b*, **104***b*, respectively, designating whether the segment is active and visible (by the code OE) or whether the segment is inactive and hidden (indicator "FF").

Figure 4.6 Cheston et al.: Page 6

4.6 Case Study: Bullwinkel et al.

U.S. Patent 6,099,317 protects "a method and system for monitoring a series of events performed in one or more applications on a computer" [8]. This patent has a state diagram that Nigon advocated [48]. The state diagram is shown in fgure 2 of U.S. Patent 6,099,317, Figure 4.7. The frst fgure of this patent uses an entire computer system in the frst fgure to show the physical means that are used in this patent as shown in Figure 4.8.

4.7 Case Study: Heizer

U.S. Patent 5,249,290 protects a server using "processes to access shared server resources in response to service requests" [22]. This patent has a functional diagram that Nigon advocated [48]. Figure 2 of U.S. Patent 5,249,290 is shown in Figure 4.9. This patent also effectively uses claims that Holmes advocated [25]. This patent has twenty claims total with f ve independent claims. The claims cover a signifcant portion of what could be covered in the claims and thus is effective in ensuring broad protection. A large number of claims does not always signify broad protection, but enough well-worded claims will ensure the proper protection. A quote of claims 1, 3, and 16 from this patent follow [22]:

> 1. A server apparatus for accessing one or more common resources using a plurality of server processes to which client service requests are assigned, said server apparatus comprising
>
> means for receiving an unassigned client service request requesting access to one of said common resources and
>
> means, responsive to a workload indication from each server process, each workload indication being less than a maximum workload for that server pro-

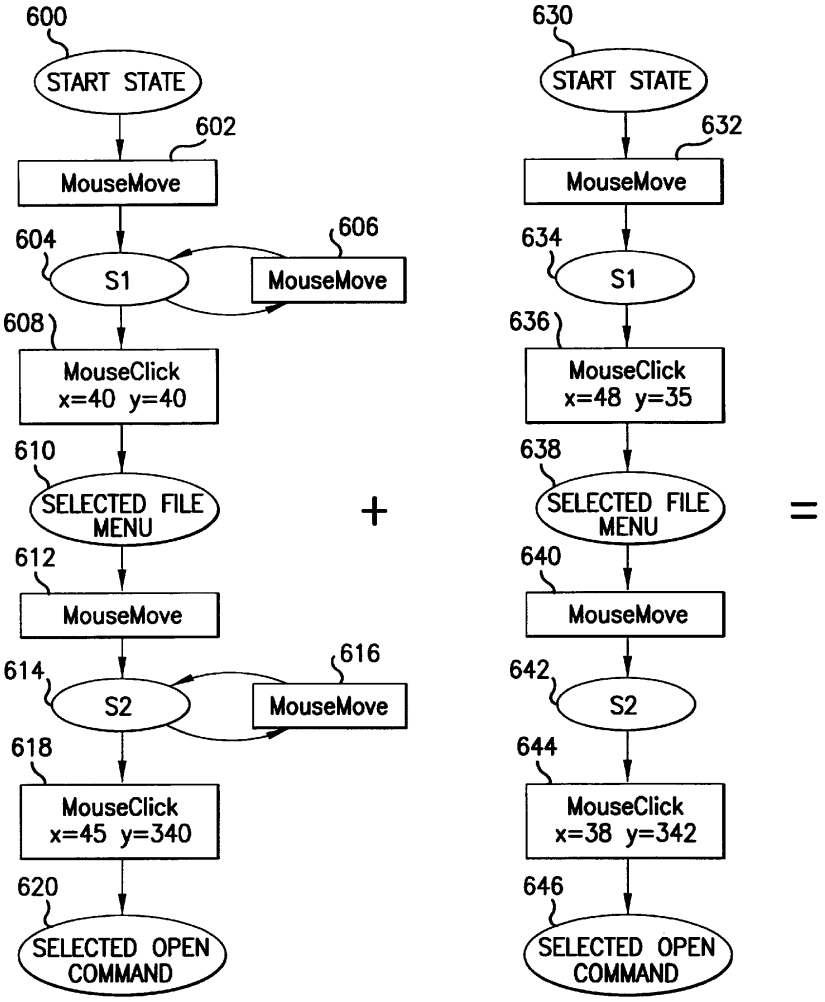

FIG.15(a) FIG.15(b)

Figure 4.7 Bullwinkel et al.: Figure 15(a-b)

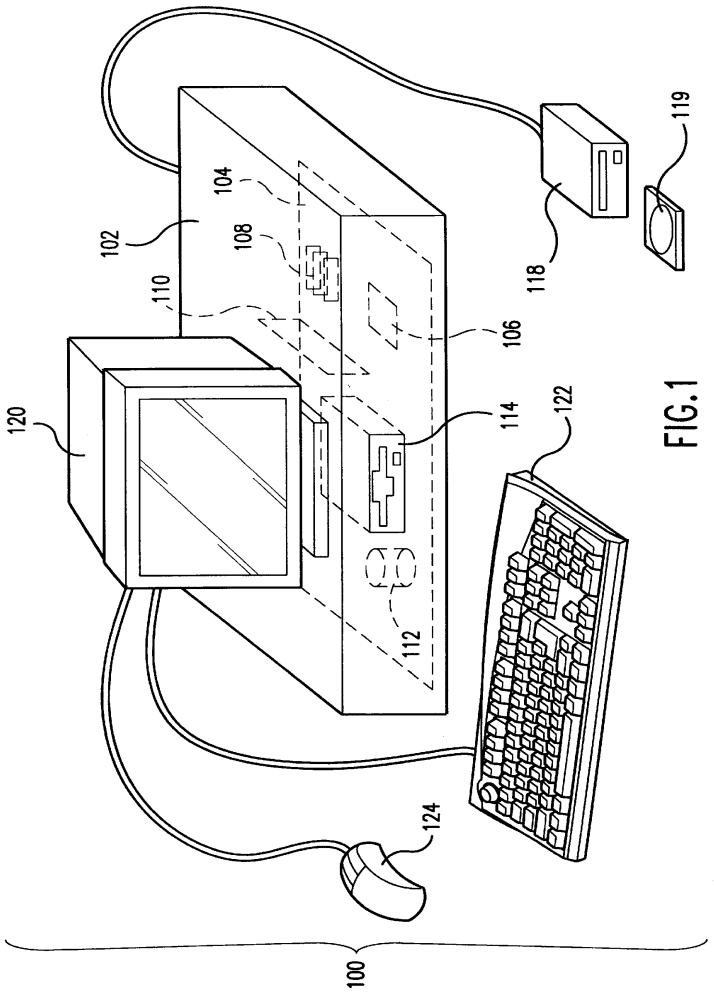

FIG. 1

Figure 4.8 Bullwinkel et al.: Figure 1

cess, for assigning said unassigned received client service request to a server process having a workload indication which is less than the workload indication of all other server processes.

3. The server apparatus of claim 1 further comprising

table means including a plurality of different predetermined workload indication ranges, each range handling a maximum number of clients and each range specifying a lower and upper limit on how many client service requests can be assigned to each server process and wherein

said assigning means, in response to a determined number of clients, selects a working range and determines the maximum number of client service requests that can be assigned to each server process.

16. A server apparatus for accessing one or more common resources using a plurality of server processes to which client service requests are assigned, said server apparatus comprising

table means, including a plurality of different predetermined workload indicator ranges, each range specifying a lower and an upper limit on how many client service requests can be assigned to each server process,

means for receiving an unassigned client service request requesting access to one of said common resources,

means, responsive to a server apparatus determined total number of client service requests, for accessing said table means to select in which range said total number of client service requests lies and thus determines, for the selected range, the number of client service requests or workload that can be assigned to each server process and

means, responsive to the selected range and a workload indicator for each server process, each workload indicator being less than the upper limit of said selected range, for assigning said unassigned received client service request to a server process having a workload indicator which is less than the workload indicator of all other server processes.

4.8 Case Study: Inoue

U.S. Patent 5,175,857 protects ”a method and apparatus for sorting object data, the object data having a data format of a next address and a record” [26]. This patent has

FIG. 2

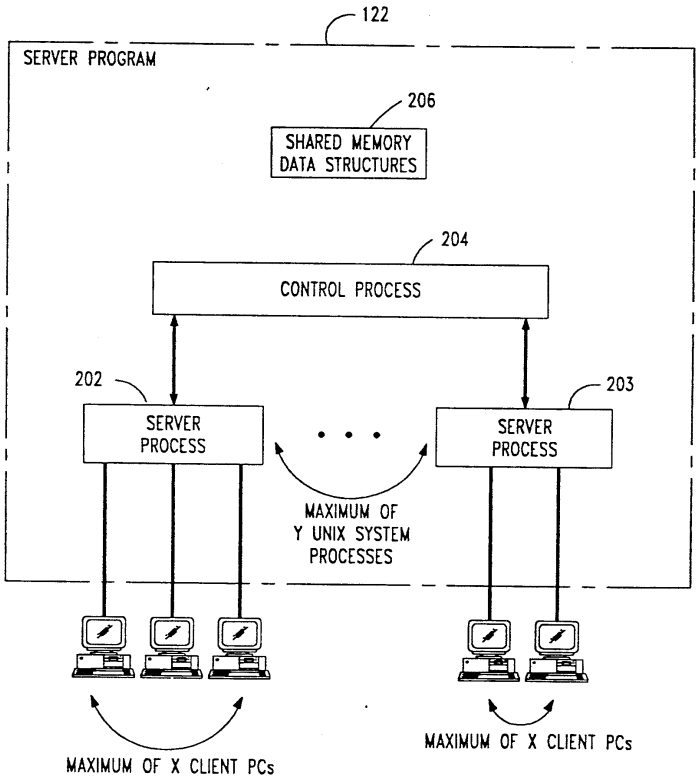

Figure 4.9 Heizer: Figure 2

a fo wchart diagram that Nigon advocated [48]. Figure 4A of U.S. Patent 5,175,857 is shown in Figure 4.10.

4.9 Case Study: Brown

U.S. Patent 4,965,765 protects “a method of distinguishing between nested expressions, functions, logic segments or other text by using a different color for each nesting level” for International Business Machines Corp. (IBM) [7]. This patent was fled on May 16, 1986 [7] which was years before graphical user interfaces (GUI) were in widespread use in computer programming applications. This indicates that this patent was used in IBM’s patent portfolio.

4.10 Case Study: Auslander et al.

U.S. Patent 4,656,583 describes ”a method for use during the optimization phase of an optimizing compiler for performing global common subexpression elimination and code motion” [2]. This patent shows the use of source as part of the specifcation that Natoli advocated [44]. The source code example is shown in Figure 4.11.

4.11 Case Study: Hellman et al.

U.S. Patent 4,200,770 protects a system that “transmits a computationally secure cryptogram over an insecure communication channel without prearrangement of a cipher key” [23]. This patent illustrates one of the early software patents that could be implemented in

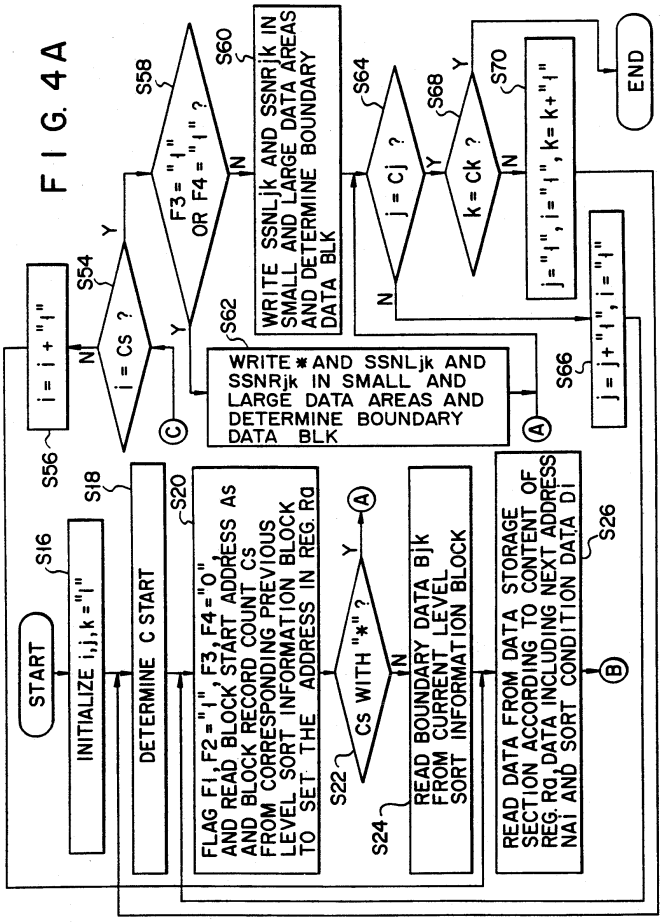

FIG. 4A

Figure 4.10 Inoue: Figure 4a

guarantee making the same instruction redundant along every control flow path entering basic block B.

The equations which define these sets of computations are as follows:

$$gux(b) = uex(b) \quad \underset{(s \in successor(b)}{gux(s)} \quad thru(b) \tag{1}$$

$$gdx(b) = dex(b) \quad \underset{p \in predecessor(b)}{\{dex(p) \quad gdx(p) \quad thru(p)\}} \tag{2}$$

$$pax(b) = \underset{p \in predecessor(b)}{\{dex(p) \quad pax(p) \quad thru(p)\}} \tag{3}$$

$$cix(b) = \underset{s \in successor(b)}{cie(s)} \tag{4}$$

$$cie(b) = \begin{array}{l} gux(b) \quad (uex(b) \quad cix(b) \quad thru((b)) \\ (pax(b) \quad (thru(b) - uex(b))) \\ \underset{p \in predecessor(b)}{(cix(p) \quad gdx(p))} \end{array} \tag{5}$$

$$insert(b) = pix(b) - gdx(b) - (pie(b) \quad thru(b)) \tag{6}$$

$$avail(b) = pie(b) \quad uex(b) \tag{7}$$

One method for solving these equations is to initialize gux(b) and pax(b) to the empty set for all basic blocks b, and to initialize gdx(b) to the set of all computations for all basic blocks except for the entry basic block, for which gdx(entry basic block) is initialized to the empty set. (The entry basic block represents the code which is executed when control first enters the program being compiled.) Equations (1), (2), and (3) are each solved by recomputing gux(b), gdx(b), and pax(b) respectively, repeatedly for all basic blocks, until recomputation does not change the sets of any of the basic blocks.

Having calculated gux(b), gdx(b), and pax(b) for all basic blocks b, we next solve equations (4) and (5) concurrently. Initialize pie(b) and pix(b) to the set of all expressions with two exceptions. pie(entry block) and pix(exit block) are initialized to the empty set. Then repeatedly recompute both pix(b) and pie(b) from equations (4) and (5) for all basic blocks b until recomputation does not change the sets for any of the basic blocks.

Finally, insert(b) and avail(b) can be computed directly from equations (6) and (7) using uex(b) and thru(b), and the results of having computed gdx(b), and pix(b) and pie(b).

The following table concisely relates to the Appendices I thru VI to the blocks of FIG. 3. These subprograms code fragments are written in PL/1 code and are sufficiently annotated as to allow any skilled programmers to utilize the present invention in an optimizing compiler in either the form shown or to easily rewrite the subprograms in another target language which would be more compatible with his compiler.

As stated previously the programs set forth in Appendices I and II represent the subject matter of the present invention as set forth in Blocks 1, 2 and 3 of FIG. 3 and the remaining Appendices II through VI represent programs which perform the operations specified in Blocks 4 thru 7 of FIG. 3.

APPENDEX REFERENCE TABLE

Appendix No.	Block of FIG. 3
1	1
2	2 & 3
3	4
4	5
5	6
6	7

APPENDIX 1

```
/* CODE FRAGMENT TO DETERMINE THE BASIS*/

$zero(seen);                  /*initialize expressions seen in basic block*/
$zero(a_basis);               /*initialize basis*/
/* Notes for reading this code:
   r is shorthand for operand(i), the i-th operand of an instruction
   x->def is shorthand for the number of results defined by the
          instruction at x
   x->use is shorthand for the number of results and input operands
          in the instruction at x
This shorthand is effective via macro instructions (not shown here)
*/

do b = b_begin to b_end;      /*walk thru every basic block*/
   do x = bl_begin(b) repeat x->il_fc until (x = bl_end(b));
              /*look at every instruction of the basic block*/
      do i = x->def + 1 to x->use;      /*look at every operand*/
         if a_basis(r) then iterate;    /*already in basis?*/
         if seen(r) then iterate;       /*already defined in block?*/
         a_basis(r) = '1'B;             /*add to basis*/
         end;
      do i = 0 to x->def;       /*Now look at all result and mark */
                                /*them as computed before used, */
         seen(r) = '1'B;     /*register that is defined*/d */
         end do i;
      end do x;
   do x = bl_end(b) repeat x->il_bc until (x = bl_begin(b));
```

Figure 4.11 Auslander et al.: Page 9

hardware and thus was allowed. An example of this hardware implementation is shown in Figure 4.12.

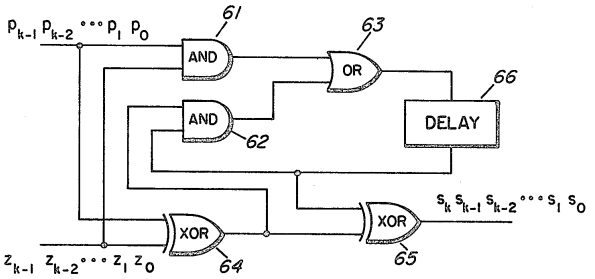

FIG. 4

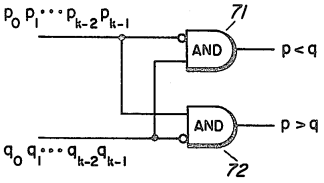

FIG. 5

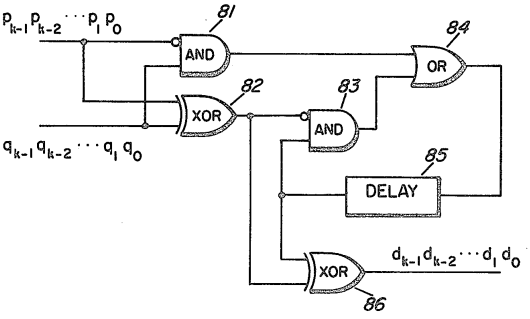

FIG. 6

Figure 4.12 Hellman et al.: Figures 4, 5, and 6

CHAPTER V

EMPIRICAL STUDY OF SOFTWARE PATENTS

This chapter contains an analysis of the research questions. Table 5.1 documents the relationship of the case studies to the research questions.

5.1 Methodology

The case studies were analyzed using cross-case analysis as described in Section 3.1. The steps of cross-case analysis were used as follows in this research [54]:

1. Data from the case studies was divided among the research questions
2. Relevant variables in the data were identifed
3. Relationships among the variables were analyzed to provide support for the research questions

5.2 Knowledge of Prior Art

Research question 1 asks,

> How does the knowledge of prior art help inventors at a university?

The case studies illustrate many of the points that were brought up in the related work and the case studies help answer this research question. Prior art searches are important. In several of the case studies (Moudgal et al. and Cheston et al.), the inventor(s) and patent

Table 5.1 Relationship of Case Studies to Research Questions

Research Questions
1. How does the knowledge of prior art help an inventor?
2. How does the understanding of patent laws and regulations help an inventor?
3. What are the differences between university and corporate environments from an inventor's viewpoint?

Case Study	Question 1	2	3	Question Remarks
Moudgal et al.	X	X		1. No prior art search by the applicant 2. Existence of prior art without adverse effects to the patent claims
Teng et al.	X			1. Extensive reference to prior art documents by the applicant
Ciacelli et al.		X	X	2. Value of using a fowchart diagram to fully disclose and protect the invention 3. Commercial companies patent technologies for their patent portfolios
Cheston et al.	X	X		1. No prior art search by the applicant 2. Use of a personal computer in the frst fgure of the patent to show the usefulness and context of the invention
Bullwinkel et al.		X	X	2. Value of using a state diagram to fully disclose and protect the invention 3. Example of what universities patent. This invention was the basis for a start-up company
Heizer		X		2. Value of using a functional diagram to fully disclose and protect the invention. Claim coverage allowed
Inoue		X		2.Illustrates the value of using a fowchart diagram to fully disclose and protect the invention
Brown			X	3. Commercial companies patent technologies for their patent portfolios
Auslander et al.		X		2. Value of inserting source code to fully disclose and protect the invention
Hellman et al.		X	X	2. Early example of a software patent 3. Example of the type of patent that universities fle

practitioner(s) obviously did not complete a prior art search before submitting the patent application to the United States Patent and Trademark Offce. These case studies are not completely consistent with the idea that a prior art search will help avoid a patent offce rejection, since one of the case studies (Moudgal et al.) that did not cite any prior art was allowed on the frst offce action. Table 5.2 summarizes the relationship between the case studies and this research question.

However, overall the case studies support the view that the knowledge of prior art helps an inventor. Only three of the ten case studies were allowed by the examiner on the frst offce action, and f ve of the ten case studies cited less then f ve prior art references. Conversely, we do not have evidence that the lack of prior art was the reason that Moudgal et al. was allowed on the frst offce action. The knowledge of prior art discussed in Section 3.5 and the evidence provided in the case studies answers this research question as shown below.

The knowledge of prior art helps an inventor at a university reduce the following risks discussed in Chapter I by avoiding or mitigating the issues below each risk:

1. Not fling a patent application
    - Infringing another patent [63]
    - Duplication of research [38]
2. Patent application not being allowed
    - Conficting prior art found during prosecution
3. Patent not holding up in court
    - Conficting prior art found after allowance

Table 5.2 Summary of Case Studies in Relation to Research Question 1

Research Question 1 How does the knowledge of prior art help an inventor?			
Case Study	References Cited	Offce Actions for Allowance	Question Remarks
Moudgal et al.	11	1	No prior art search by the applicant
Teng et al.	42	6	Extensive reference to prior art documents by the applicant
Ciacelli et al.	17	1	Extensive reference to prior art documents by the applicant
Cheston et al.	10	2	No prior art search by the applicant
Bullwinkel et al.	4	2	No prior art cited by examiner
Heizer	3	4	No prior art cited by examiner
Inoue	21	3	Extensive reference to prior art documents by the applicant. No prior art cited by examiner
Brown	3	4	No prior art cited by examiner
Auslander et al.	1	3	No prior art cited by examiner
Hellman et al.	2	1	No prior art cited by examiner

5.3 Understanding of Patent Laws and Regulations

Research question 2 asks,

> How does the understanding of patent laws and regulations help inventors at a university?

The complexity of the patent laws and regulations do not allow inventors to be experts in the area, but, as illustrated in the case studies and described in Sections 3.3 and 3.4, a knowledge of the laws and regulations allows an inventor to understand what patent practitioners require to prosecute a patent application. The case studies are from a wide range of areas in the software feld, but they all use the same fundamental sections and types of fgures to describe the invention. The sections of a patent include a title, a cross-reference to related applications, statements regarding federally funded research, background, brief summary, brief description of the drawings, detailed description, claim(s), abstract, and drawings [60]. The types of fgures that can be used include functional block, fo wchart, and state diagrams [48]. The knowledge of these sections and the types of fgures allows an inventor to relay the pertinent information to the patent practitioner.

5.3.1 Readability

One important aspect of patents is the readability. In general, patents need to be easy to read and understand so that someone like a judge or jury can understand the basic concept of the patented technology [57]. Increased readability also helps to fulfll the three requirements in 35 U.S.C. §112, frst paragraph, described in Section 3.3.

Table 5.3 shows readability information for the case studies. The grade level in Table 5.3 refers to the level at which the text is rated for the case studies. For example, if the grade level is ten then a tenth grader should be able to understand the text. Since the language that patents are written in is legalese and highly technical in nature, the patents range from thirteen to sixteen in grade level. The sentence complexity refers to the length of the sentences, with one hundred being the most complex. The vocabulary complexity refers to the length of the words in the text, with one hundred being the most complex. The grade level information of the patents is shown in Figure 5.1, and the complexity level is shown in Figure 5.2.

Burge suggests in Section 3.3 that a patent application should "set forth the pith of the invention in terms a grade-school student can grasp" [9]. None of the case studies meet this challenge by Burge as shown in Figure 5.1. Overall, the readability of the case studies is at a high grade level (i.e. poor readability). If an inventor understood the reasoning behind making patents easier to read, then he or she could justify spending the time and effort in making the patent application easier to read and understand.

5.3.2 Diagram Complexity

Another important aspect of patent understandability is the use of and complexity of diagrams. The use of simple diagrams comes from 35 U.S.C. §112, frst paragraph, described in Section 3.3. Nigon explains the value of using diagrams to describe a software invention in Section 3.4[48].

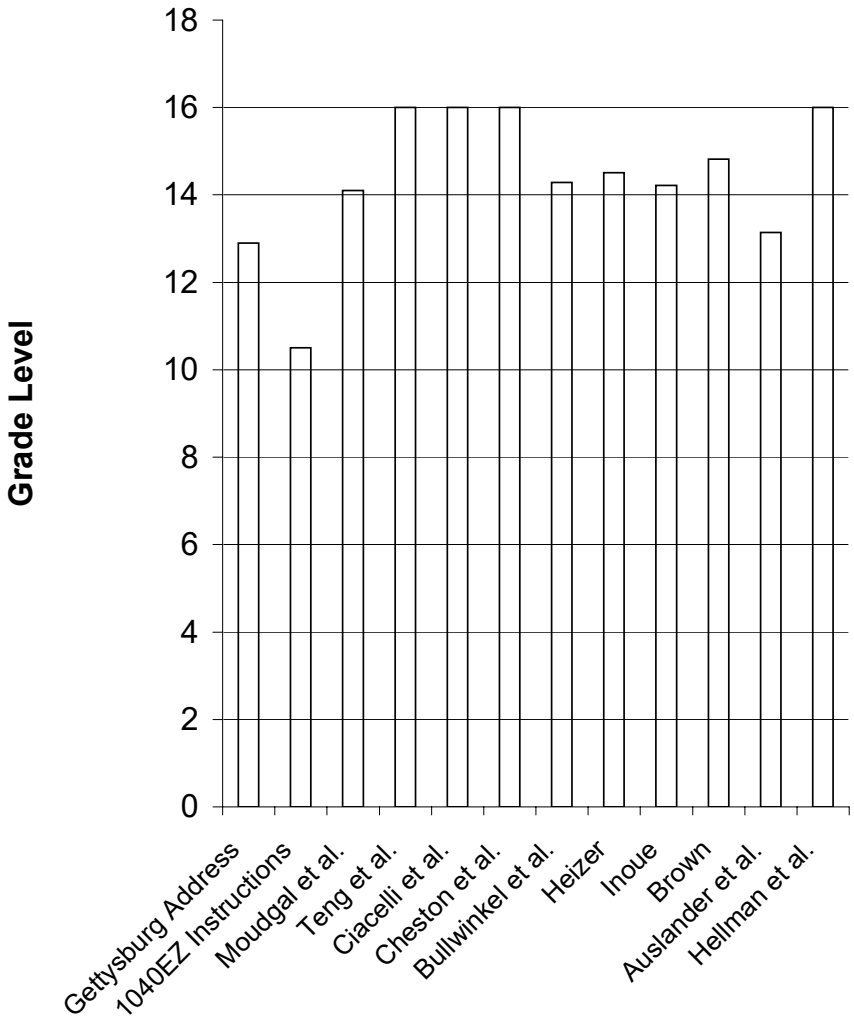

Figure 5.1 Case Study Grade Level Graph

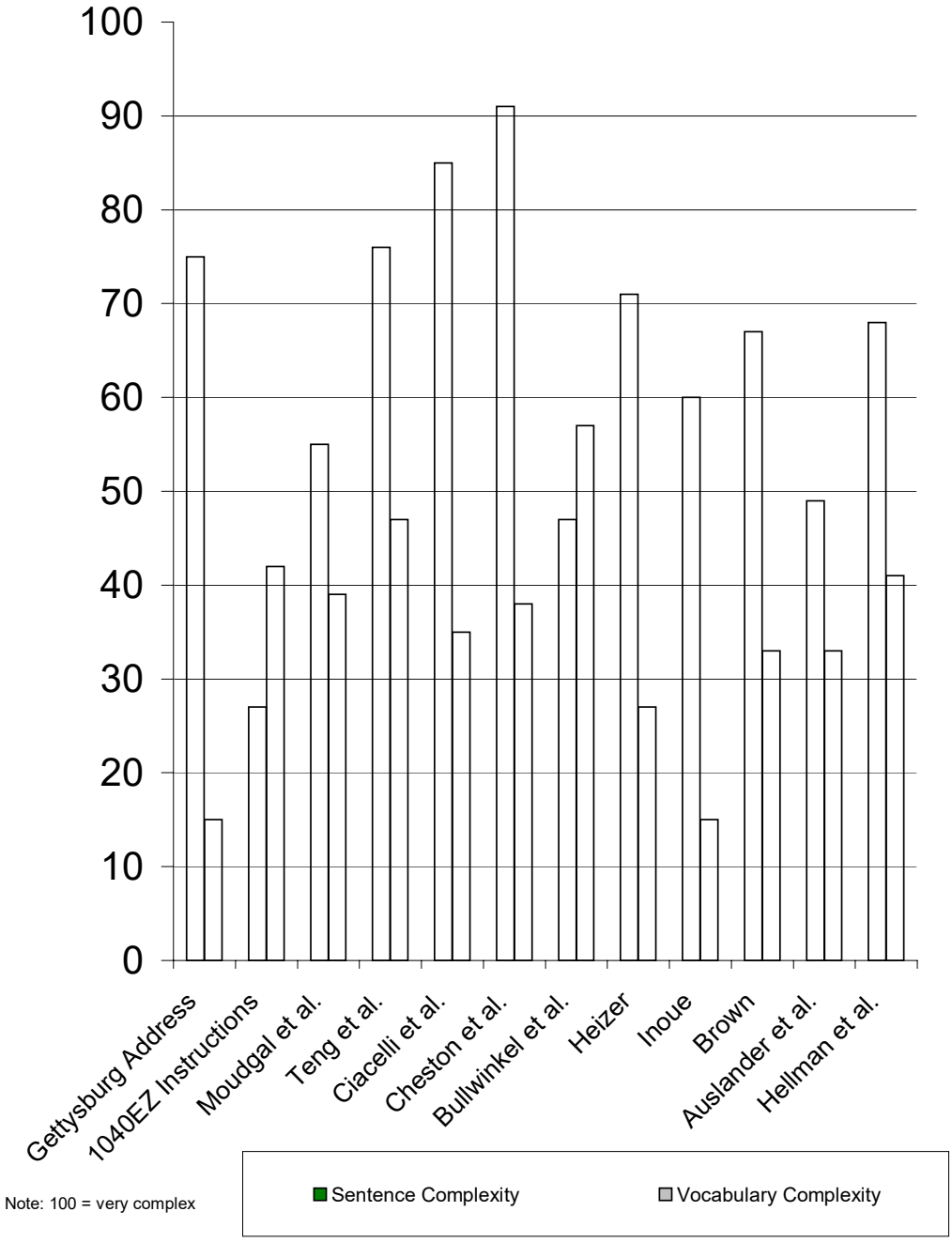

Figure 5.2 Case Study Complexity Graph

Table 5.3 Case Study Readability

U.S. Patent	Grade Level	Sentence complexity*	Vocabulary complexity*
Gettysburg Address	12.9	75	15
1040EZ Instructions	10.5	27	42
Moudgal et al.	14.1	55	39
Teng et al.	16.0	76	47
Ciacelli et al.	16.0	85	35
Cheston et al.	16.0	91	38
Bullwinkel et al.	14.3	47	57
Heizer	14.5	71	27
Inoue	14.2	60	15
Brown	14.8	67	33
Auslander et al.	13.1	49	33
Hellman et al.	16.0	68	41

* 100=very complex

Diagrams help fulfll the requirements in 35 U.S.C. §112, frst paragraph, and simple diagrams make it easier to understand the invention. The case studies support the use of simple diagrams, as can be seen in Table 5.4, which shows the cyclomatic complexity of the diagrams. Cyclomatic complexity is “the number of linearly independent paths” [39]. The independent paths can thus be an indication of the complexity of a diagram, because when a low number of paths exist then a diagram is easier to understand.

The information in Table 5.4 shows that the most complex fgure has a complexity of nine. One fgure with a complexity of nine is shown in Figure 5.3 and Figure 5.4. This fgure has two parts, 4a and 4b, that are considered one when calculating the cyclomatic complexity, since both fgures would be considered a single aspect of the invention. This method of calculation was used for the fgures listed in Table 5.4. Overall, the use of

diagrams in the case studies is high, and the complexity of the diagrams is low, which helps answer this research question. The understanding of the patent laws and regulations allows inventors to grasp the importance of the full disclosure of their invention utilizing diagrams of low complexity as shown in the case studies.

Table 5.4 Case Study Diagram Complexity

Case Studies	Figures															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Moudgal et al.		5	2	4			9					2				
Teng et al.		1	2	1	5	1	1									
Ciacelli et al.	2	4	3	3												
Cheston et al.		3														
Bullwinkel et al.					4	3	2	2	5	2	2	2	1	1	5	6
Heizer			5		2											
Inoue			2	9	6											
Brown	7															
Auslander et al.	1	1	1													
Hellman et al.	9			6	3	6										

5.3.3 One-Year Bar

University software inventors have to work with university technology licensing offces. The goal of these offces is to protect and license university inventions. In contrast, the inventor wants to publish his research results. Inventors from universities who are faculty have job advancement motivations to publish. The confict of duties between the university's technology licensing offce and the inventor's desire to publish creates a bur-

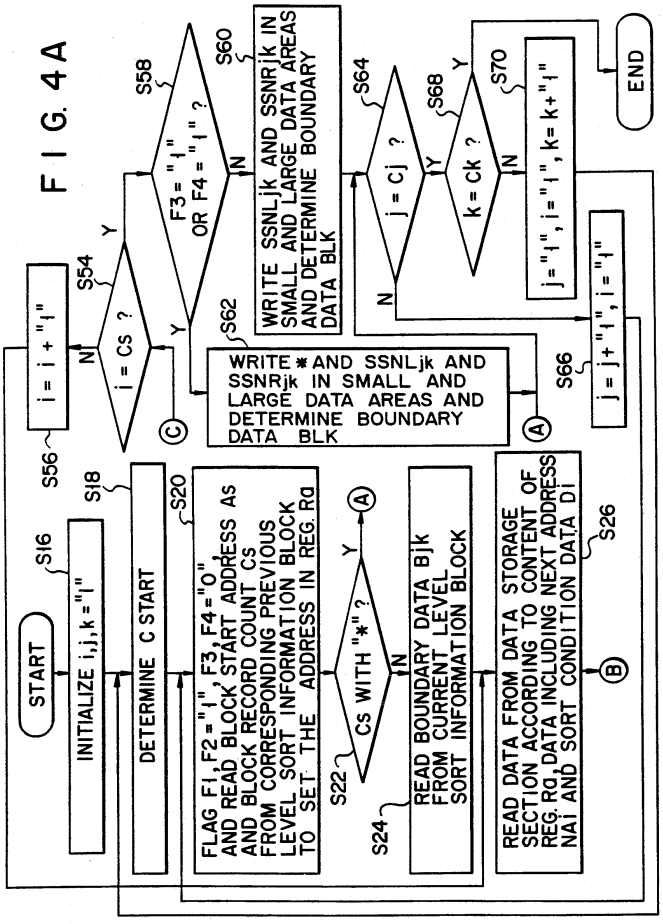

FIG. 4A

Figure 5.3 Inoue: Figure 4a

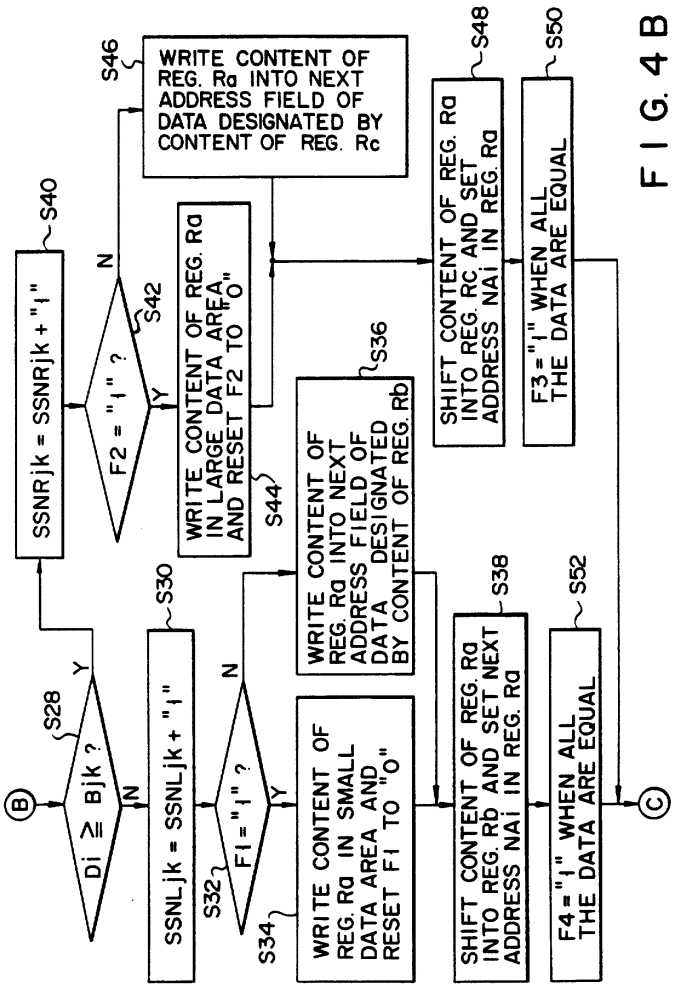

FIG. 4B

Figure 5.4 Inoue: Figure 4b

den on the inventor. University inventors have to balance their publication productivity and the desire by their university to license inventions that could be proftable because of the "one-year bar" on fling a patent application. If an inventor publishes, sells, or discloses to the public his or her invention, then one year from the disclosure date is the application deadline for a U.S. patent [60].

Although none of the case studies provide evidence for meeting fling deadlines in the United States, it is well-recognized that papers are the products of academic research. Papers that are published more than one year prior to the fling date of a U.S. application can "bar a patent under 35 U.S.C. 102(b)" [60]. This one-year bar is important for university inventors to understand and work to avoid. The understanding of the patent laws and regulations is important for inventors to understand because of the possible loss of patent rights after publication or public use.

5.3.4 Analysis

The basics of U.S. patent law and regulations are important for inventors to understand. The issues surrounding the readability, diagram construction, and one-year bar in patent preparation and prosecution answers this research question.

The understanding of patent laws and regulations helps inventors at universities reduce the following risks discussed in Chapter I by avoiding or mitigating the issues below each risk:

1. Not fling a patent application

- One-year bar after the invention is published, sold, or disclosed to the public [60]

2. Patent application not being allowed

   - Invention disclosure does not fully describe invention, thus conception date could be lost in interference proceedings
   - Application not complying with 35 U.S.C. §112, frst paragraph [57]

3. Patent not holding up in court

   - Diagrams are not understandable by juror and/or judge [57]
   - Patent specifcation is not understandable by juror and/or judge [57]
   - Inventor not disclosing bar date to patent attorney until lawsuit

5.4 Differences between University and Corporate Environments

Research question 3 asks,

> What are the differences between university and corporate environments from an inventor's viewpoint?

Software patents at a university are far and few between because of the open academic environment in which projects are developed. To acquire patent protection an invention must be developed in some degree of secrecy. Most software developed at universities has been protected using copyright. University researchers are not doing research for commercial gain like commercial researchers, but they are doing research for academic purposes [53]. This difference in research purposes is a signifcant change for inventors who solicit corporate funding for their research funding since the inventors might not be able to publish his or her research results.

5.4.1 Embryonic Technology

Universities, unlike commercial companies, do not have the resources or for-proft drive to protect all possible inventions [59]. Much of embryonic software technology at universities does not receive funding for patent protection because of the lack of need for a patent portfolio [53]. Several of the case studies illustrate examples of companies fling patents on early stage technology.

The Ciacelli et al. case study is one example of a company fling a patent on idea while it is still embryonic [11]. This patent illustrates the beneft a company receives by patenting an invention. International Business Machines Corporation (IBM) fled this patent on June 24, 1997 before encrypted data for copyright protection was widely used in commercial products. IBM now has a broad patent on a technology that might be widely used in the future. IBM highlights this patent as an important software patent in the security feld that is in its patent portfolio on its website [27]. The Brown case study is another example of a patent to be placed in IBM's patent portfolio [7]. This patent was fled on May 16, 1986 before extensive graphical user interfaces were used on most computer systems.

5.4.2 Bayh-Dole Act

Since the Bayh-Dole Act in 1980, universities have had the option of retaining the intellectual property rights of inventions that came out of federally funded research [13]. This option opened the door for universities to collect revenue from thousands of licenses. Some of the obligations that universities have under the Bayh-Dole Act follow [13]:

- Disclose each new invention to the responsible federal funding agency
- Have written agreements with faculty and technical staff
- Government has right to use the invention
- Periodic reports to the funding agency
- Share a portion of licensing revenue with inventor(s)

The Bayh-Dole Act has affected the way universities operate by requiring them to manage the inventions originating from federal funding [59]. These obligations affect inventors by requiring them to disclose inventions to their university's technology licensing offce. The licensing revenue obligation is a beneft to university inventors. It is important to understand the signifcant difference between university and corporate environments because of the legal requirements associated with the Bayh-Dole Act. Unlike corporations, universities are obligated to report inventions and therefore must have the support from the inventors to accomplish this task.

5.4.3 Ownership

Universities, like corporations, usually retain ownership of anything that was created or conceived on their time or using their facilities [41]. Thus, most research done at a university is turned over to the university. However, there is one big difference between most universities and corporations. Most universities provide a royalty-sharing mechanism for the inventors as required by the Bayh-Dole Act [13]. This royalty-sharing mechanism usually returns 40% to 50% of the royalty income back to the inventors thus encouraging them to disclose before publishing [41]. The knowledge of this royalty-sharing mechanism is

thus important for university inventors. The knowledge of ownership rights, the royalty-sharing mechanism, and the related work discussed in Section 3.6 answers this research question. This difference of the royalty-sharing mechanism is a signifcant difference between university and corporate environments that is important for inventors to understand because of the fnancial implications.

5.4.4 Analysis

Since corporations are in business to make money, most of the time their strategy is to patent any variety of an invention that could be important to the business. This strategy is different from the strategy that universities follow. The differences between corporate and university environments is important for inventors to understand because of the consequences in the corporate environment of not quickly identifying and protecting inventions [59]. One advantage for inventors in the university environment is the royalty-sharing mechanism required by the Bayh-Dole Act. Most companies do not share royalties with its inventors, thus inventors at a university have a fnancial advantage over inventors in the corporate environment. The differences in how embryonic technology is approached, in the obligations under the Bayh-Dole Act, how ownership at universities is different, and the related work in Sections 3.6, 3.8, and 3.9 answers this research question.

The understanding of the differences between university and corporate environments helps inventors at universities reduce the following risks discussed in Chapter I by avoiding or mitigating the issues below each risk:

1. Not fling a patent application

    - Research funding from corporations with provisions that do not allow publications or patents
    - Research funding from the federal government requiring permission to fle patent
    - Inadequate funding for patent preparation and prosecution

2. Patent application not being allowed

    - Inadequate funding for patent prosecution

3. Patent not holding up in court

    - Inadequate funding for defense of patent

CHAPTER VI

CONCLUSIONS

This chapter provides the reasoning of how the research questions support the hypothesis, contributions of this research, and further research that could be considered based on the completed research.

The risks associated with university inventions are as follows:

1. Not fling a patent application
2. Patent application not being allowed
3. Patent not holding up in court

The following are the research questions designed to provide evidence for or against the hypothesis.

1. How does the knowledge of prior art help inventors at a university?
2. How does the understanding of patent laws and regulations help inventors at a university?
3. What are the differences between a university and corporate environments from an inventor's viewpoint?

The relationships between the risks and the research questions along with the issues and resolutions associated with the research questions are shown in Table 6.1, Table 6.2, and Table 6.3. The resolutions to the issues help answer the research questions and provide evidence for the hypothesis.

Table 6.1 Relationship of Risk 1 to the Research Questions

Risk 1: Not fling a patent application

Research Questions
1. How does the knowledge of prior art help an inventor?
2. How does the understanding of patent laws and regulations help an inventor?
3. What are the differences between university and corporate environments from an inventor's viewpoint?

Question	Issue	Resolution
1	Infringing another patent	By understanding how the knowledge of prior art can help an inventor, an inventor at a university can complete a prior art search before starting new research and thus avoid infringing another patent.
	Duplication of research	By understanding how the knowledge of prior art can help an inventor, an inventor at a university can complete a prior art search before starting new research and thus avoid duplicating research already completed.
2	One-year bar after the invention is published, sold, or disclosed to the public	By understanding the patent laws and regulations, an inventor can avoid the one-year bar and ensure that his or her invention can be patented.
3	Research funding from corporations with provisions that do not allow publications or patents	By understanding the differences between university and corporate environments, an inventor can avoid this issue by ensuring that the research grants do not disallow any publications.
	Research funding from the federal government requiring permission to fle patent	By understanding the differences between university and corporate environments, an inventor can promptly disclose any federal funding associated with his or her invention and avoid this issue, by allowing adequate time to disclose the invention to the federal funding agency.
	Inadequate funding for patent preparation and prosecution	By understanding the differences between university and corporate environments, an inventor can understand why this issue might occur and work to gain early funding for the patent preparation and prosecution.

Table 6.2 Relationship of Risk 2 to the Research Questions

Risk 2: Patent application not being allowed

Research Questions
1. How does the knowledge of prior art help an inventor?
2. How does the understanding of patent laws and regulations help an inventor?
3. What are the differences between university and corporate environments from an inventor's viewpoint?

Question	Issue	Resolution
1	Conficting prior art found during prosecution	By understanding how the knowledge of prior art can help an inventor, an inventor at a university can complete a prior art search before helping the patent practitioner prepare the patent application and thus avoid this issue.
2	Invention disclosure does not fully describe invention, thus conception date could be lost in interference proceedings	By understanding the patent laws and regulations, an inventor can avoid this issue by ensuring that his or her invention disclosure fully describes the invention.
	Application not complying with 35 U.S.C. §112, frst paragraph	By understanding the patent laws and regulations, an inventor can avoid this issue by fully describing his or her invention by utilizing diagrams and making the patent easy to read and understand.
3	Inadequate funding for patent prosecution	By understanding the differences between university and corporate environments, an inventor can avoid this issue by ensuring adequate funding is available for the patent prosecution.

Table 6.3 Relationship of Risk 3 to the Research Questions

Risk 3: Patent not holding up in court

Research Questions
1. How does the knowledge of prior art help an inventor?
2. How does the understanding of patent laws and regulations help an inventor?
3. What are the differences between university and corporate environments from an inventor's viewpoint?

Question	Issue	Resolution
1	Conficting prior art found after allowance	By understanding how the knowledge of prior art can help an inventor, an inventor at a university can complete a prior art search before helping the patent practitioner prepare the patent application and thus avoid this issue.
2	Diagrams are not understandable by juror and/or judge	By understanding the patent laws and regulations, an inventor can use diagrams that are easy to understand and thus avoid this issue.
	Patent specifcation is not understandable by juror and/or judge	By understanding the patent laws and regulations, an inventor can help the patent practitioner write a patent application that is easy for any judge and juror to understand.
	Inventor not disclosing bar date to patent attorney until lawsuit	By understanding the patent laws and regulations, an inventor will realize what details to convey to the patent practitioner and avoid this issue.
3	Inadequate funding for defense of patent	By understanding the differences between university and corporate environments, an inventor can work to license his or her technology, so that the university does not have to attempt to defend his or her patent.

The hypothesis of this research is that:

> An inventor at a university can improve the protection of his or her software patent by applying certain information about patent prosecution practices and the relevant prior art.

Overall, the evidence associated with the risks that answers each of the research questions provides a frm basis for the buildup of evidence that supports this hypothesis. Each research question addresses a specifc issue in the hypothesis. The patent prosecution practices information comes directly from research questions 2 and 3. The relevant prior art information comes directly from research question 1. Thus, the hypothesis is supported by "the weight of evidence" in the preceding chapters [54].

6.1 Contributions

This thesis answers questions associated with the patent preparation and prosecution process. These answers could be utilized by inventors at universities to protect their software inventions and avoid the risks discussed above. Software inventors at universities will be able to use the information analyzed in this thesis to gain a better understanding of software patent protection and how it can be applied to their inventions.

6.2 For Further Research

Further research includes an analysis of the fle wrappers of the case studies to determine how the knowledge of prior art played in the patent prosecution. Further research also includes a long-term study of inventors at a university. One group would be inventors

who understand and have applied the principles explained in this thesis and the other group would be those who have not been exposed to the principles explained in this thesis.

REFERENCES

[1] G. Aharonian, "Does the Patent Offce respect the software community?," *IEEE Software*, vol. 16, no. 4, Jul./Aug. 1999, pp. 87–89.

[2] M. A. Auslander, J. Cocke, and P. W. Markstein, *Method for Improving Global Common Subexpressions Elimination and Code Motion in an Optimizing Compiler*, US patent 4,656,583, to International Business Machines Corporation, Patent and Trademark Offce, Washington, D.C., 1987.

[3] M. J. Balconi-Lamica, "Writing an Invention Disclosure," *Successful Patents and Patenting for Engineers and Scientists*, M. A. Lechter, E. C. Clifford, R. B. Famiglio, and R. J. Joenk, eds., IEEE Press, New York, 1995, chapter 16, pp. 185–191.

[4] F. Blakemore, "Patenting computer software," *Engineering Management Journal*, vol. 9, no. 4, Aug. 1999, pp. 157–160.

[5] J. Boyle, *Shamans, software, and spleens : law and construction of the information society*, Harvard University Press, Cambridge, Massachusetts, 1997.

[6] A. Bragg, "Software Patents," *IT Professional*, vol. 3, no. 4, Jul./Aug. 2001, pp. 39–42.

[7] M. R. Brown, *Distinguishing Nested Structures by Color*, US patent 4,965,765, to International Business Machines Corporation, Patent and Trademark Offce, Washington, D.C., 1990.

[8] R. A. Bullwinkel, J. S. Calhoun, and J. W. Griffn, *Device that Interacts with Target Applications*, US patent 6,099,317, to Mississippi State University, Patent and Trademark Offce, Washington, D.C., 2000.

[9] D. A. Burge, *Patent and Trademark Tactics and Practice*, 2nd edition, John Wiley & Sons, New York, 1984.

[10] R. W. Cheston, H. Locker, and D. B. Rhoades, *Data Processing System and Method for Recovering from System Crashes*, US patent 6,195,695, to International Business Machines Corporation, Patent and Trademark Offce, Washington, D.C., 2001.

[11] M. L. Clacelli, J. W. Urda, W. M. Lam, J. L. Kouloheris, and J. E. Fetkovich, *Apparatus, Method and Computer Program Product for Protecting Copyright Data Within*

a Computer System, US patent 6,236,727, to International Business Machines Corporation, Patent and Trademark Offce, Washington, D.C., 2001.

[12] Committee on Science U.S. House of Representatives, *Patent System and Modern Technology Needs: Meeting the Challenge of the 21st Century*. U.S. Government Printing Offce, June 1996.

[13] Council On Governmental Relations, "A Guide To The Law And Implementing Regulations," *The Bayh-Dole Act*, 1999, http://www.cogr.edu/bayh-dole.htm (current 10 Nov. 2002).

[14] R. Davis, P. Samuelson, M. Kapor, and J. Reichman, "A new view of intellectual property and software," *Communications of the ACM*, vol. 39, no. 3, Mar. 1996, pp. 21–30.

[15] A. L. Durham, *Patent Law Essentials: A Concise Guide*, Quorum Books, Westport, CT, 1999.

[16] M. Duvall, "Patent Pushers: Sell, Not Sue," *Interactive Week*, vol. 6, no. 28, July 1999, p. 7.

[17] S. J. Frank, "Balancing Act," *IEEE Spectrum*, vol. 39, no. 6, June 2002, pp. 65–66.

[18] B. A. Galler, *Software and Intellectual Property Protection: Copyright and Patent Issues for Computer and Legal Professionals*, Quorum Books, Westport, CT, 1995.

[19] S. L. Garfnk el, R. M. Stallman, and M. Kapor, "Why Patents Are Bad for Software," *Issues in Science and Technology*, vol. 8, no. 1, Fall 1991, pp. 50–55.

[20] T. D. Harris III, *The Legal Guide to Computer Software Protection: A Practical Handbook on Copyrights, Trademarks, Publishing, and Trade Secrets*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1985.

[21] D. L. Hayes, "Apple v. Microsoft: Under a Microscope," *The Computer Lawyer*, vol. 11, no. 2, Feb. 1994, pp. 1–15.

[22] I. J. Heizer, *Method of and Apparatus for Operating a Client/Server Computer Network*, US patent 5,249,290, to AT&T Bell Laboratories, Patent and Trademark Offce, Washington, D.C., 1993.

[23] M. E. Hellman, B. W. Diffe, and R. C. Merkle, *Cryptographic Apparatus and Method*, US patent 4,200,770, to Stanford University, Patent and Trademark Offce, Washington, D.C., 1980.

[24] M. R. Henry, M. K. Cho, M. A. Weaver, and J. F. Merz, "DNA Patenting and Licensing," *Science*, vol. 297, no. 5585, Aug. 2002, p. 1279.

[25] W. N. Holmes, "The Evitability of Software Patents," *Computer*, vol. 33, no. 3, Mar. 2000, pp. 30–34.

[26] S. Inoue, *System for Sorting Records having Sorted Strings each having a Plurality of Linked Elements each Element Storing Next Record Address*, US patent 5,175,857, to Kabushiki Kaisha Toshiba, Patent and Trademark Offce, Washington, D.C., 1992.

[27] International Business Machines Corporation, "Patents: Software," *Intellectual Property and Licensing*, http://www.ibm.com/ibm/licensing/patents/software.shtml (current 18 Nov. 2002).

[28] International Business Machines Corporation, "Software," *Intellectual Property and Licensing*, http://www.ibm.com/ibm/licensing/portfolio/software.html (current 13 Feb. 2002).

[29] G. Irlam, "Examples of Software Patents," http://www.base.com/software-patents/examples.html (current 13 Feb. 2002).

[30] P. C. Judge, "From the patent wars, a patent medicine," *Business Week*, , no. 3579, May 1998, p. 111.

[31] B. Kahin, "The Software Patent Crisis," *Technology Review*, vol. 93, no. 3, Apr. 1990, pp. 52–59.

[32] P. D. Kelly, "Keeping Research Records," *Successful Patents and Patenting for Engineers and Scientists*, M. A. Lechter, E. C. Clifford, R. B. Famiglio, and R. J. Joenk, eds., IEEE Press, New York, 1995, chapter 15, pp. 156–184.

[33] G. J. Kirsch, "The Changing Roles of Patent and Copyright Protection for Software," *Legal Information for Internet Professionals*, http://www.gigalaw.com/articles/2000-all/kirsch-2000-04-all.html (current 5 Nov. 2002).

[34] G. J. Kirsch, "Software Protection: Patents versus Copyrights," *Legal Information for Internet Professionals*, http://www.gigalaw.com/articles/2000-all/kirsch-2000-03-all.html (current 5 Nov. 2002).

[35] G. J. Kirsch and Y. Skulikaris, "Intellectual Property Protection for Software in the United States and Europe: The Changing roles of Patents and Copyrights," *Proceedings: 2000 International Conference on Software Engineering*, Limerick, Ireland, June 2000, Association of Computing Machinery, p. 827.

[36] K. Ku, "Software Licensing: Stanford's Approach," *Computing Research News*, vol. 14, no. 1, Jan. 2002, pp. 3–8.

[37] M. A. Lechter, E. C. Clifford, R. B. Famiglio, and R. J. Joenk, eds., *Successful Patents and Patenting for Engineers and Scientists*, IEEE Press, New York, 1995.

[38] D. C. Marcus, "Benefts of using Patent Databases as a Source of Information," *Successful Patents and Patenting for Engineers and Scientists*, M. A. Lechter, E. C. Clifford, R. B. Famiglio, and R. J. Joenk, eds., IEEE Press, New York, 1995, chapter 6, pp. 65–70.

[39] T. J. McCabe, "A Complexity Measure," *IEEE Transactions on Software Engineering*, vol. SE-2, no. 4, Dec. 1976, pp. 308–320.

[40] A. R. Miller and M. H. Davis, *Intellectual Property: Patents, Trademarks, and Copyright in a Nutshell*, third edition, West Group, St. Paul, MN, 2000.

[41] Mississippi State University, "MSU Policy 7006," *Policy and procedure statement on intellectual property*, http://www.msstate.edu/dept/audit/7006.html (current 5 Nov. 2002).

[42] Mississippi State University: Offce of Intellectual Property and Technology Licensing, "Applying for Patent Protection at MSU," http://iptl.msstate.edu/process.html (current 5 Nov. 2002).

[43] A. N. Moudgal, B. M. Kuttanna, and A. Tzeng, *Apparatus and Method for Preventing Cache Data Eviction During an Atomic Operation*, US patent 6,347,360, to Sun Microsystems, Inc., Patent and Trademark Offce, Washington, D.C., 2002.

[44] A. J. Natoli, "Software patents: beating a dead horse and other important concerns," *Professional Program Proceedings: Electro '96*, Somerset, New Jersey, Apr./May 1996, IEEE Region I, pp. 269–276.

[45] F. W. Neitzke, *A Software Law Primer*, Van Nostrand Reinhold Company, New York, 1984.

[46] K. Nichols, "The Age of Software Patents," *IEEE Computer*, vol. 32, no. 4, Apr. 1999, pp. 25–31.

[47] K. Nicols, *Inventing software : the rise of "computer-related" patents*, Quorum Books, Westport, CT, 1998.

[48] K. Nigon, "Drafting the Specifcation," *Electronic and Software Patents: Law and Practice*, S. W. Lundberg and S. C. Durant, eds., BNA Books, Washington, DC, 2000, chapter 5, pp. 201–216.

[49] L. S. Nixon and J. S. Davidson, "Software can be patented," *IEEE Potentials*, vol. 16, no. 4, Oct. 1997, pp. 21–22.

[50] T. O'Reilly, "The Internet Patent Land Grab," *Communications of the ACM*, vol. 43, no. 6, June 2000, pp. 29–31.

[51] D. Pressman and R. Stim, *Nolo's patents for beginners*, Nolo, Berkeley, California, 2000.

[52] P. Samuelson, "Is copyright law steering the right course?," *IEEE Software*, vol. 5, no. 5, Sept. 1988, pp. 78–86.

[53] P. R. Savage, "Betting on a New Idea," *IEEE Spectrum*, vol. 39, no. 6, June 2002, pp. 63–64.

[54] C. B. Seaman, "Qualitative Methods in Empirical Studies of Software Engineering," *IEEE Transactions on Software Engineering*, vol. 25, no. 4, Jul./Aug. 1999, pp. 557–572.

[55] R. Stern, "Micro Law-software patents," *IEEE Micro*, vol. 10, no. 2, Apr. 1990, pp. 8–11.

[56] R. Stern, "The PTO on software patents," *IEEE Micro*, vol. 15, no. 4, Aug. 1995, pp. 2–3,77–78.

[57] G. A. Stobbs, *Software Patents*, Wiley Law Publications, New York, 1995.

[58] C.-C. Teng and B. Jahromi, *Computer Network*, US patent 6,327,045, to Microsoft Corporation, Patent and Trademark Offce, Washington, D.C., 2001.

[59] B. Thorner, "University Technology Transfer Practices," 1998, http://articles.corporate.fndla w.com/articles/fle/frms/bph/bp h000047 (current 10 Nov. 2002).

[60] United States Patent and Trademark Offce, *Manual of Patent Examining Procedure*, eighth edition, United States Government Printing Offce, Washington, D.C., August 2001.

[61] U.S. Department of Commerce, *Patents and How to Get One A Practical Handbook*, Dover Publications, Inc., Mineola, New York, 2000.

[62] W. Wade, *Patent Guide for the Research Director*, Advance House Publishers, Ardmore, Pennsylvania, 1964.

[63] T. A. Wiens, "Searching Software Inventions," *Electronic and Software Patents: Law and Practice*, S. W. Lundberg and S. C. Durant, eds., BNA Books, Washington, DC, 2000, chapter 3, pp. 61–86.

[64] A. Wolfe, "Patent nonsense?," *Electronic Engineering Times*, , no. 840, Mar. 1995, pp. 70–72.